E.T.S. de Ingeniería Industrial,
Informática y de Telecomunicación

Control domótico de una vivienda con un Smartphone mediante comunicación vía wifi

Grado en Ingeniería
en Tecnologías Industriales

Trabajo Fin de Grado

Autor: Echeveste Zayas, Javier
Tutor: Andueza Unánua, Ángel María
Pamplona, 27 de junio de 2016

upna Universidad Pública de Navarra
Nafarroako Unibertsitate Publikoa

Resumen

El objetivo de este proyecto es el diseño y construcción del control domótico de una vivienda con un Smartphone mediante comunicación vía Wifi.

Se realizará una App mediante la cual se pueda interactuar con el Arduino que controlará la iluminación, temperatura y el consumo de electricidad de la vivienda. Para ello se diseñará un Shield con el que se acondicionará la señal de los sensores de efecto hall y temperatura (NTC). Dicha fase de diseño incluirá la búsqueda y selección de los componentes adecuados para después proceder a su volcado en una placa de circuito impreso.

Una vez se ha fabricado la placa y la App se comunicarán ambos dispositivos para que se sincronicen las variables necesarias en una red local vía wifi..

Por último se conectarán cuatro bombillas simulando la iluminación de distintas estancias de la vivienda y otra bombilla simulando la caldera a unos relés que son activados por el Arduino.

Palabras Claves

- Arduino Yún
- Domótica
- Red Local
- Arduino Shield

Índice

1. JUSTIFICACIÓN Y OBJETIVOS

1.1. Limitaciones

En general el uso de sistemas domóticos en los hogares españoles no está muy extendido. Esto es debido entre otros motivos a diferentes limitaciones en la oferta de los sistemas domóticos convencionales. Las principales problemas que han llevado a que la industria domótica no despegue se encuentran, entre otros, en:

- El precio del sistema domótico: Una vivienda domótica puede llegar a costar desde los 2.000€ hasta la cifras mucho mayores que uno quiera. Aunque a simple vista parezca una cantidad importante, se estima que la infraestructura básica para domotizar una vivienda, sólo encarece el precio un 2 % de media. Pero aproximando a lo que se quiere hacer en este proyecto una empresa de domótica podría cobrar alrededor de 2.000€ [4]

  El problema es que en una vivienda ya construida es mucho más caro instalar el sistema domótico debido a que habría que cambiar o adaptar gran parte de la vivienda.

- Complejidad: En muchos de los casos el usuario recibe el sistema domótico y éste no es muy predictivo o incluso puede llegar a ser complicado según para qué usuario.

- Sistemas cerrados: Uno de los mayores problemas de la domótica es que los sistemas domóticos son sistemas cerrados. Es decir, son como unas cajas cerradas. Se instala un sistema en el cual los usuarios no saben cómo trabaja el sistema ni pueden acceder a él. Por tanto si se desea modificar el sistema, añadir algún componente más el usuario es incapaz de hacerlo y tiene que volver a contratar a algún especialista.

- Otras: Se podrían comentar algunas que otras limitaciones o impedimentos que ha hecho que no se pueda desarrollar como en años anteriores se esperaba. Pueden ser desconocimiento a la domótica o la indiferencia en el mercado de la construcción que son incapaz de ver e incluir en su oferta viviendas que ofrezcan las numerosas ventajas de la domótica como recurso de valor añadido.

1.2. Objetivos

En este Trabajo Fin de Grado se elaborará un proyecto donde se haga un *Control domótico de una vivienda con un Smartphone mediante comunicación vía wifi*. Los objetivos de este proyecto por tanto, vienen dados por las anteriores limitaciones.

- **Monitorización y control de la iluminación y temperatura de la vivienda basado en la plataforma Arduino.**
- **Control de la energía consumida y aproximación de las facturas mensuales.**
- **Creación de una interfaz con el usuario sencilla, ordenada e intuitiva.**
- **Desarrollo de un sistema económicamente barato y ampliable en el futuro.**
- **Actuación sobre diferentes elementos de la planta o vivienda.**

2. INTRODUCCION

2.1. Domótica: Definición

El trabajo que se expone en este documento está basado en la domótica. Según la Real Academia Española la palabra domótica tiene el siguiente significado: "*Conjunto de sistemas que automatizan las diferentes instalaciones de una vivienda*".

El término domótica viene de la unión de las palabras *domus* (que significa casa en latín) y *tica* (de automática, palabra en griego, 'que funciona por sí sola'). Por tanto, se puede entender la domótica como el conjunto de tecnologías aplicadas al control y la automatización inteligente de la vivienda, que permite una gestión eficiente del uso de la energía, que aporta seguridad y confort, además de comunicación entre el usuario y el sistema.

2.2. Domótica en España

En España La asociación Española de Domótica (CEDOM) es la única Asociación a nivel nacional que reúne a todos los agentes del sector de la Domótica/Inmótica en España: fabricantes de sistemas domóticos, equipos auxiliares, distribuidores, integradores, instaladores, centros tecnológicos y de formación, universidades y medios de comunicación. En el estudio de mercado elaborado por CEDOM entre los años 2012-2014, destaca el crecimiento del **15,12%** registrado en el último ejercicio. Los fabricantes de estas tecnologías han facturado un volumen de **40 millones de euros** de negocio en 2014. A pesar de esta subida, no se ha alcanzado la cifra de 45,2 millones que se obtuvo en el año 2012.

Hay que comentar también que en dicho estudio no se tiene en cuenta la distribución ni la instalación de los equipos domóticos, por lo que el volumen de facturación del sector podría llegar a ser muy superior al previamente descrito.

Finalmente, teniendo en cuenta que la seguridad del hogar es, sin lugar a dudas, la función más valorada de la vivienda domótica según sus usuarios, su introducción en nuestro país sería bastante sencilla; según los datos del Ministerio del Interior se producen más de nueve robos por hora en las casas españolas, siguiendo un incremento alarmante en los últimos años. A la vista de estos resultados se puede afirmar que la domótica en España es un sector importante y en pleno auge.

2.3. Elementos de una vivienda domótica

Las instalaciones domóticas están formadas por diferentes elementos o dispositivos. Dichos dispositivos pueden resumirse en los siguientes:

2.3.1. La pasarela residencial

La pasarela residencial es el dispositivo frontera entre las distintas redes de acceso externas y las redes internas del edificio inteligente[2]. Estas pasarelas son las que se encargan de toda la gestión de servicios internos, adaptación de los protocolos utilizados por los distintos dispositivos a todos los niveles así como La gestión de todos los dispositivos internos de forma local o remota.

Las pasarelas pueden ser pasarelas residenciales de banda ancha como routers o hubs, adaptados entre los datos de la red interna de la vivienda y la conexión externa. Por otro lado estas pasarelas también pueden ser pasarelas residenciales multiservicios. Estas son más complejas y potentes que los anteriores ya que proporcionan varios interfaces para redes de datos y control con diferentes tecnologías.

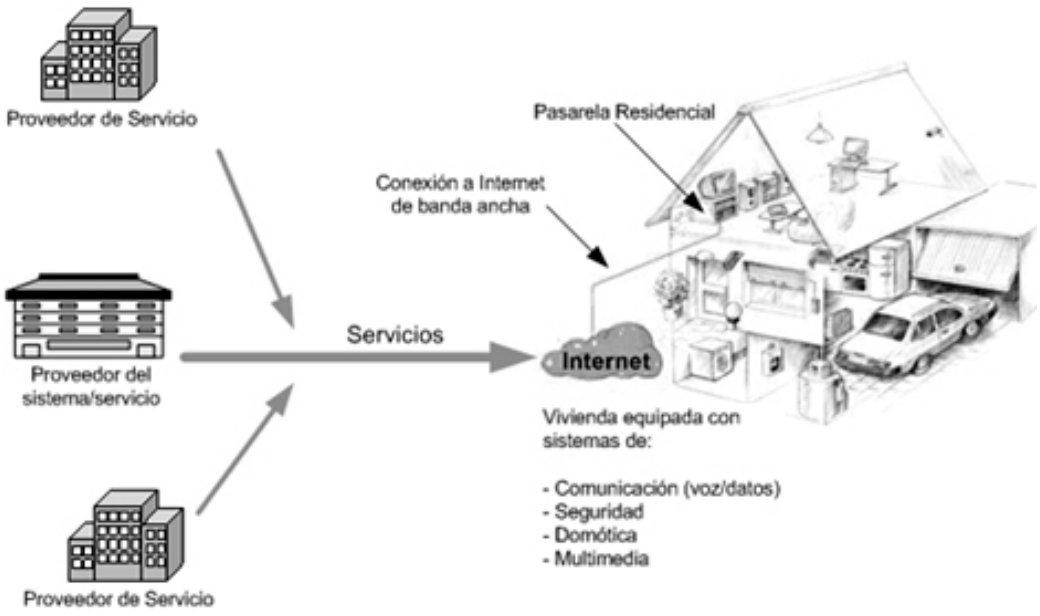


Imagen 1. Esquema de redes de una vivienda domótica

2.3.2. Los sensores

Los sensores son una parte vital de la vivienda domótica. Un sensor es un objeto capaz de detectar magnitudes físicas o químicas, llamadas variables de instrumentación, y transformarlas en variables eléctricas. Los sensores pueden ser de muy diferentes tipos. Hay sensores que miden posición, velocidad, otros fuerza otros presión, luminosidad…

Imagen 2. Sensor de movimiento

2.3.3. Los actuadores

Un actuador es un dispositivo capaz de transformar energía hidráulica, neumática o eléctrica en la activación de un proceso con la finalidad de generar un efecto sobre un proceso automatizado. Los actuadores pueden se de tipo electrónico, hidráulico, neumático y eléctrico, pudiendo modificar el estado de ciertos equipos o instalaciones dentro de la vivienda (el aumento o la disminución de la calefacción o el aire acondicionado, el corte del suministro de gas o agua, el envío de una alarma a una centralita de seguridad, etc.).

Imagen 3. Actuados neumático

2.3.4. Controlador

El controlador es un cerebro electrónico encargado de recoger toda la información proporcionada por los sensores distribuidos en los distintos puntos de control de la vivienda, procesarla, y generar las órdenes que ejecutarán los actuadores.

2.4. Arquitecturas

Los sistemas domóticos pueden ser diseñados de diferentes arquitecturas. Cada una de ellas tiene sus ventajas y sus desventajas por lo que cada una de ellas encajará mejor en una situación que en otra dependiendo del uso final. Las arquitecturas principales son las mencionadas a continuación.[1]

2.4.1. Centralizada

Este tipo de arquitecturas está basado en un controlador centralizado que recibe información de múltiples sensores y, una vez procesada, genera las órdenes oportunas para los actuadores.

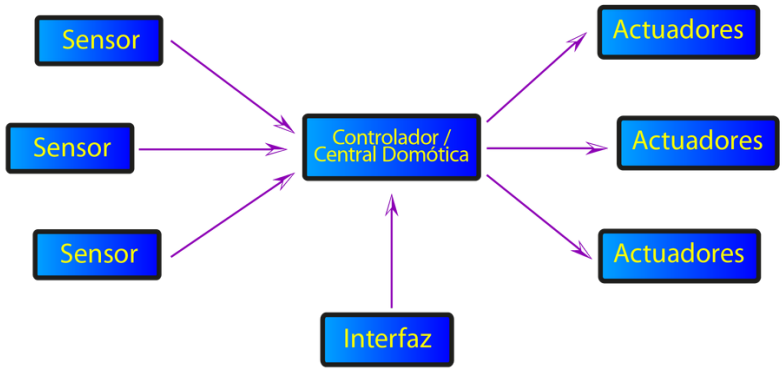


Imagen 4. Esquema de una arquitectura centralizada

Las ventajas al usar este tipo de arquitecturas son varias;

- Se evita la redundancia, la inconsistencia, el procesamiento de los datos ofrece un mejor rendimiento y resulta más fiable que los sistemas distribuidos.

- Fácil reemplazamiento. Por otro lado si se estropea un sensor o un actuador es muy sencillo cambiarlo. Solo tienen una conexión con el controlador.

- Crecimiento. Se pueden conectar gran cantidad de sensores y actuadores.

En cambio las desventajas al utilizar este tipo de arquitectura son las siguientes:

- Controlador delicado. Si el controlador sufre algún tipo de daño, el sistema deja de servir.

- Crecimiento. Si lo que se desea es tener un sistema muy extenso con muchos sensores y muchos actuadores y que se puedan en un futuro conectar más, es mejor utilizar otro tipo de sistemas.

2.4.2. Descentralizado

En el tipo de arquitectura descentralizado, toda la inteligencia del sistema está distribuida por diferentes módulos. Se trata de unas arquitecturas centralizadas donde los controladores son conectados en serie entre sí.

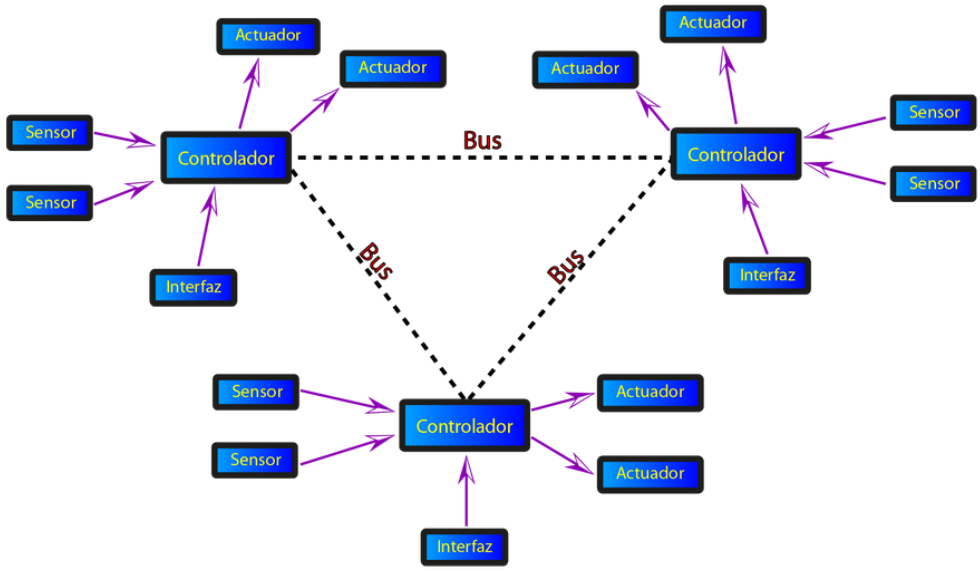


Imagen 5. Esquema de una arquitectura descentralizada.

Las ventajas de utilizar una arquitectura de este tipo son:

- División de tareas: Al no estar centralizado hay diferentes controladores por lo que no es necesario que toda la información la tenga un controlador.

- Velocidad de respuesta rápida: debido a que los controladores están menos saturados que en la arquitectura centralizada.

Las desventajas de utilizar una arquitectura de este tipo son:

- Si la distribución de los actuadores, sensores etc… está mal hecha puede ser peor que un sistema centralizado.

- Es más caro (tanto la instalación como el mantenimiento) y tiene mayor complejidad que el sistema centralizado.

2.4.3. Distribuida

En el tipo de arquitectura distribuida, en cambio, son los propios sensores y actuadores los que tienen la inteligencia y son capaces de decidir. Todos ellos están conectados mediante un bus central.

Las ventajas de utilizar una arquitectura de este tipo son:

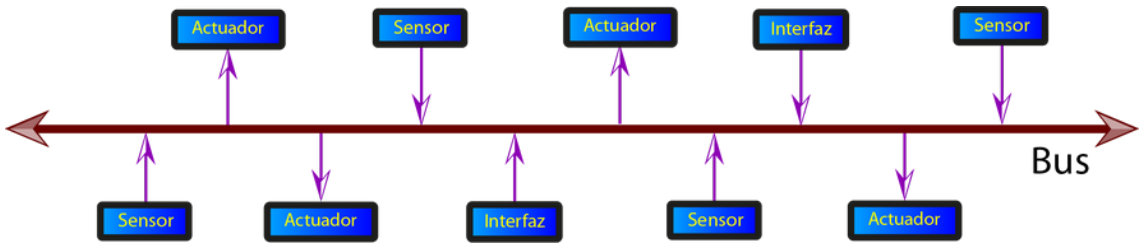


Imagen 6. Esquema de una arquitectura distribuida.

Las ventajas de utilizar una arquitectura de este tipo son:

- Respuesta rápida debido a que el propio sensor o actuador tiene su propia inteligencia, lo que evita la perdida de tiempo por comunicación. El mismo sensor toma la decisión de actuar o dejar de actuar sobre el sistema.

- Funcionalidad más amplia. Se puede obtener información a través de monitoreo, telecontrol…

- Inherente al crecimiento. Se pueden conectar todos los actuadores o sensores que se deseen.

Las desventajas de utilizar una arquitectura de este tipo son:

- Requerimientos de mayores controles de procesamiento.

- Velocidad de propagación de información. Si se desea mandar información hay veces que se colapsa debido a que todos los sensores quieren transmitir en el mismo momento. Por tanto, a veces pueden llegar a ser conexiones muy lentas.

- El coste es elevado debido a que los sensores y actuadores lleven su propio controlador. No se utiliza cualquier sensor ni cualquier actuador.

2.5. Aplicaciones de la domótica en la vida real

La domótica puede ser muy útil para la sociedad a pesar de que se piense que ésta solo es utilizada por personas con cierta discapacidad. La domótica presta diferentes servicios al usuario en diferentes ámbitos en la vida de una persona independientemente de su condición física.

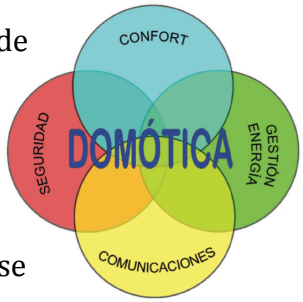


Imagen 7. Aplicaciones de la domótica.

Estos servicios ofrecidos por la domótica pueden agruparse en cinco aspectos o ámbitos principales:

2.5.1. Programación y ahorro energético

En estos últimos años se ha revolucionado todo lo relacionado con el ahorro energético. La causa de esta revolución viene definida por otros aspectos.

Por un lado hay que mencionar la crisis que llegó en el año 2007. Con el ahorro energético se pretende **optimizar el gasto** de los recursos para la misma acción mediante una gestión eficiente. Ello conlleva una necesidad menor de energía lo que se traduce en un menor importe en la factura mensual. En este ámbito las acciones más comunes sueles ser:

- Climatización y calderas: Se pretende zonificar la vivienda. Es decir, si se desea calentar una zona o estancia de la casa, se calienta esa zona dejando las otras como previamente estaban.

- Control de toldos y persianas: Se detecta cuando hay viento o cuando hay mucho sol mediante diferentes sensores para que los actuadores actúen sobre ellos.

- Gestión eléctrica: Se pretende no gastar lo que no se necesita. Por ejemplo si el sistema detecta que hay luces encendidas en una zona o estancia de la casa donde no hay nadie, éstas se apagarán

2.5.2. Confort

El confort conlleva todas las actuaciones que se puedan llevar a cabo que mejoren el **bienestar o comodidad** en una vivienda.

- Iluminación: En lo referente a la iluminación del apartamento se puede automatizar el apagado o encendido de cada punto de luz, así como la regulación de la iluminación dependiendo del nivel de luz que haya en el exterior.

- Automatización: Se puede automatizar todos los distintos sistemas, o instalaciones dotándolos de control eficiente y de fácil manejo para el usuario.

- Gestión y control: Gracias a la tecnología de hoy en día se puede monitorizar, gestionar y controlar todo desde una App de un Smartphone o cualquier dispositivo electrónico y controlar remotamente desde este mediante Internet.

2.5.3. Seguridad

La seguridad de una vivienda es primordial. Por eso las viviendas domóticas tienen como vital el aspecto de la seguridad. El sistema domótico se dota de sensores de modo que si alguien entra en alguna habitación él mismo avisa a la policía y cierra las puertas y ventanas para que el ladrón no pueda salir. Además puede transmitir imágenes al usuario de lo que está pasando dentro de su propiedad.

Por otro lado también existen, detectores de incendios, de inundación de concentración de monóxido de carbono etc. Debido a todo esto la **seguridad de la vivienda incrementa** exponencialmente gracias al uso del sistema domótico.

2.5.4. Comunicaciones

En cuanto a comunicaciones se tienen múltiples canales con los que el sistema puede comunicarse directamente con el usuario o con otro tipo de servicios. Los sistemas o infraestructuras de comunicaciones que posee el hogar son:

- Control remoto: desde Internet para controlar los sistemas activos.

- Teleasistencia: servicio, dirigido a personas mayores que viven solas o a personas con discapacidad, que permite pedir ayuda en caso de urgencia, desde el propio domicilio.

- Telemantenimiento: permite realizar a distancia las mismas acciones que ejecutaríamos conectados con un cable directo a nuestros equipos de automatización. Así, sin necesidad de moverse desde se puede reprogramar, actualizar, visualizar… el equipo remoto.

- Transmisión de alarmas: Cuando una alarma suena o cuando se detecta cualquier anomalía se transmite dichas alarmas e incluso imágenes al usuario y a los servicios de seguridad en caso de que fueran necesarios.

2.5.5. Accesibilidad

Se podría incluir en el ámbito del confort o de las comunicaciones. Bajo este ámbito se incluyen las aplicaciones o instalaciones de control remoto del entorno que favorecen la autonomía personal de personas con discapacidad o limitaciones.

Se prioriza favorecer un diseño accesible para suplir limitaciones funcionales de las personas, incluyendo las personas discapacitadas o mayores. El objetivo no es que las personas con discapacidad puedan acceder a estas tecnologías, porque las tecnologías en si no son un objetivo, sino un medio. El objetivo de estas tecnologías es favorecer la autonomía personal. Los destinatarios de estas tecnologías son todas las personas, ya que por enfermedad, discapacidad o envejecimiento.[3]

3. CONTEXTO TECNOLÓGICO

En este apartado se introducirá la tecnología que se ha usado en este proyecto.

3.1. Descripción de la vivienda

En este subapartado se explicará cómo es la vivienda en la que el sistema domótico está instalado así como el sistema domótico. Evidentemente la vivienda descrita en este proyecto será una vivienda ficticia debido a que el sistema domótico no será instalada en dicha vivienda.

La vivienda constará de un pasillo central, y a un lado la cocina, y al otro el baño. Por último al final del pasillo estará el salón. Se quiere que en cada habitación o estancia del hogar se pueda **controlar la iluminación** mediante una App de Android.

Por otro lado, también se quiere controlar (gracias a la App) la **temperatura de la casa y la calefacción**. También se desea hacer una aproximación de la **energía consumida** durante el mes y hacer una aproximación económica del coste de la factura de la luz..

En la vivienda también existen obviamente habitaciones, pero éstas no son de gran importancia debido a que en este proyecto no se contempla ninguna acción en ellas. Por tanto en este proyecto principalmente se tratará el control del baño, la cocina, el salón y el pasillo.

3.2. Instalación domótica

Como se ha descrito en el apartado anterior es de vital importancia entender qué es el Arduino, como funciona la App del Smartphone y cómo es la comunicación entre ellos.

Para hacerse una idea general, el usuario gracias a la App podrá enviar órdenes al controlador del sistema (Arduino en este caso) y el controlador hará que el sistema actúe. Primero la App se conectará con el Arduino y preguntará por el estado de las variables (luces, temperatura, energía…). Una vez se sabe el estado de éstas, si llega una acción (de la App) se cambiará la variable y se actualizará. Si no hay ninguna acción también se deberá verificar. Esto se debe a que el usuario también mediante el interruptor o termostato en la propia vivienda puede cambiar el estado de las luces o de la temperatura deseada, respectivamente; por tanto cambia el estado de la variable.

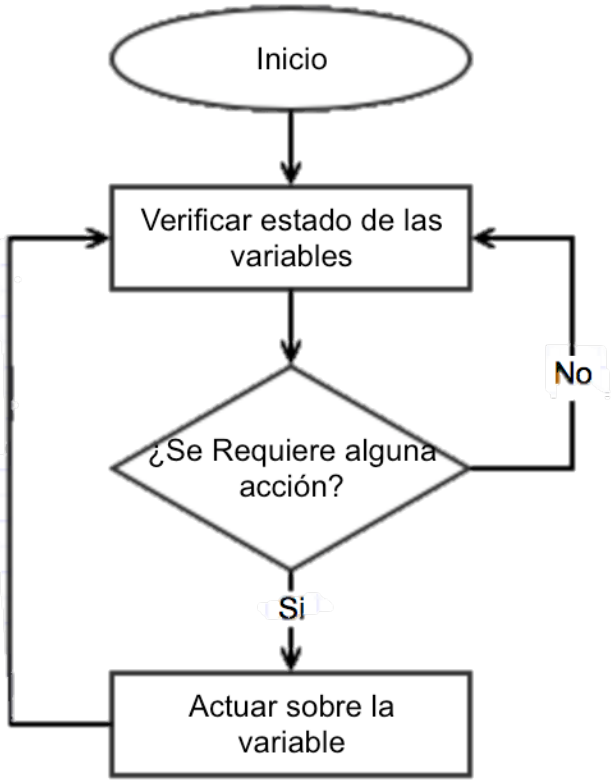


Imagen 8. Diagrama de flujo del sistema domótico

Por ejemplo, si se quiere apagar alguna luz, mediante la App se apaga mediante el *switch* como se muestra en la Imagen 10, y se manda la instrucción de que se apague al Arduino. El Arduino lo entenderá y hará que gracias a un relé se ejecute la acción deseada.

Imagen 9. Switch de encendido/apagado de la App

3.2.1. Controlador del sistema

3.2.1.1. ¿ Qué es un controlador?

Un controlador de dispositivo o manejador de dispositivo es un programa informático que permite al sistema operativo interaccionar con un periférico, haciendo una abstracción del hardware y proporcionando una interfaz (posiblemente estandarizada) para utilizar el dispositivo. Es una pieza esencial del software, sin la cual el hardware sería inutilizable. [5]

Para el proyecto que se está describiendo se desea un controlador de hardware libre. Este tipo de controladores son aquellos dispositivos de hardware cuyas especificaciones y diagramas esquemáticos son de acceso público, ya sea bajo algún tipo de pago, o de forma gratuita. La filosofía del software libre es aplicable a la del hardware libre, y por eso forma parte de la cultura libre.

Existen muchísimos tipos de controladores de hardware libre. Los más populares son Arduino y Raspberry Pi aunque realmente existen algunas más como Open conmute Project, Uzebox, Rep Rap, etc.

3.2.1.2. Arduino como controlador

Para el presente proyecto un objetivo fundamental es que desarrollar un sistema económicamente barato y que sea sencillo de utilizar. Para ello, se debe elegir un controlador que sea sencillo de usar y que el usuario pueda conocer de antemano. Los controladores más conocidos son Arduino y Raspberry Pi. Por tanto, por su precio y por su sencillez se debe elegir entre uno de estos dos controladores. A continuación se intentará analizar y comparar cada uno de ellos para finalmente elegir uno de ellos como controlador del sistema.

La **ventaja de Arduino frente a Raspberry Pi** es que es excelente para crear diferentes proyectos de domótica porque cuenta con diferentes entradas y salidas para conectar un sinfín de sensores y actuadores de forma clara y sencilla. La plataforma Raspberry Pi es más compleja para el uso de sensores o actuadores.

La **ventaja de Raspberry frente a Arduino**, en cambio, es que la velocidad de la placa en la Raspberry Pi es mucho mayor (700MHz) que en la placa de Arduino (16MHz). Por otro lado, la placa de Arduino es básicamente un microcontrolador con el que podemos conectar nuestro ordenador directamente y programar diferentes funciones para sus sensores. En cambio, la placa de Raspberry Pi es un microprocesador o, lo que es lo mismo, un ordenador que dispone de 256 o 512 MB de memoria RAM y una distribución Linux como sistema operativo. [6]

Cabe destacar que el precio entre los dos es muy parecido, aunque el de la Raspberry Pi es ligeramente mayor.

Por tanto se elige la plataforma de **Arduino** para el uso como controlador en este proyecto debido a la clara ventaja del uso de sensores y actuadores de una plataforma y la otra. La velocidad de uno u otro realmente no afectará mucho al sistema debido a que no se necesita mucha precisión. En otras palabras, no es de gran importancia si una acción se realiza instantáneamente o si tarda medio segundo en llevarse a cabo.

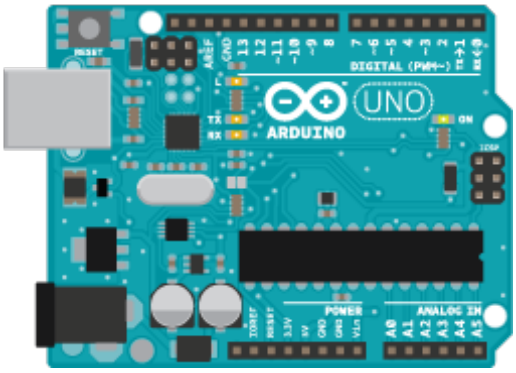


Imagen 10. Vista superior de la placa Arduino Uno

3.2.1.3. *Arduino en este proyecto*

La plataforma Arduino cuenta con una muy amplia gama de placas cada una con sus diferentes características que hacen para cada proyecto más especial a una o a otra. De los muchos tipos de Arduinos existentes para este proyecto se ha elegido el modelo **Arduino Yun**.

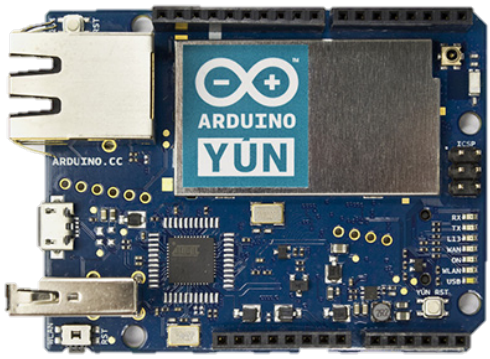


Imagen 11. Vista superior del Arduino Yun

Esta placa además de todas las ventajas que tiene la plataforma de Arduino tiene otras como la conexión inalámbrica a internet o un pequeño sistema operativo linux .El Arduino Yun se conecta a la señal Wi-fi de casa y crea una LAN (local área network) gracias al sistema operativo, por lo que se puede conectar con el router de la vivienda y acceder a Arduino.

3.2.2. App Smartphone

Se va a hacer una App que sirva para controlar el sistema domótico. Mediante dicha App se podrá mandar las ordenes oportunas para poder controlar la iluminación del hogar, termostato de la casa y demás.

La App será únicamente para Android debido a que es una de las plataformas más usada por los usuarios y, por tanto mas accesible o que más gente pueda disfrutarla. Según se dijo en el subapartado *2.2. Objetivos* es primordial que la interfaz de la App sea sencilla, ordenada e intuitiva logrando así un manejo más eficiente de la App sin apenas conocimientos de un Smartphone.

La plataforma con la que se programará la aplicación se llama **Android Studio**. Android Studio es un entorno de desarrollo integrado para la plataforma Android. Fue anunciado el 16 de mayo de 2013 en la conferencia Google I/O, y reemplazó a Eclipse como el IDE oficial para el desarrollo de aplicaciones para Android. La primera versión estable fue publicada en diciembre de 2014. [7] La programación en este entorno se hace usando el lenguaje java.

Por tanto, al ser una plataforma desarrollada para Android por Google (quien compró dicha plataforma en 2005) es la herramienta ideal con la que desarrollar la App.

3.2.3. Comunicación

La comunicación es uno de los pilares fundamentales del sistema domótico que se pretende usar. El Smartphone con su App debe estar permanentemente comunicándose con Arduino. Esta comunicación puede ser de las los siguientes maneras: LAN o Mediante Internet.

3.2.3.1. LAN

Un modo de conexión es en modo LAN (Local Area Network). Tanto el dispositivo móvil como el Arduino se encuentran dentro de la misma red Wi-fi. La conexión entre ellas no va más allá del router y éste se encarga de comunicarlas. Dentro del LAN cada dispositivo tiene su propia dirección IP privada con lo que es muy fácil comunicarlas. Cabe destacar que el router no necesariamente tiene que estar conectado a Internet.

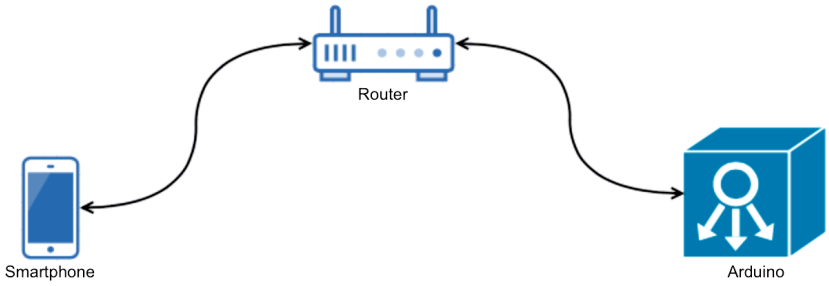


Imagen 12. Diagrama de conexión LAN

3.2.3.2. Mediante Internet

El otro modo que se ha contemplado para este proyecto es la conexión mediante Internet.

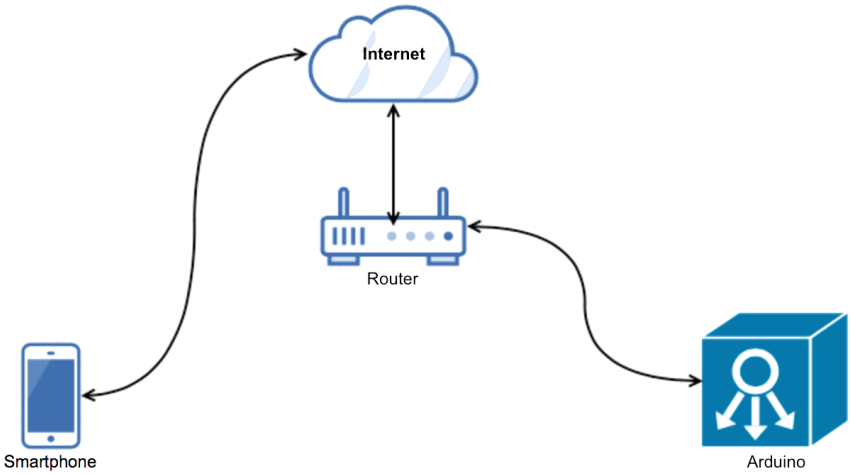


Imagen 13. Diagrama de conexión mediante Internet

En este modo el Smartphone podría estar conectado a cualquier red. Como el router está también conectado a internet se pueden conectar el Smartphone con el router gracias a Internet mediante IP públicas asignadas al router y al telefono.

3.2.3.3. Opción elegida

La red Local la crea el router y a cada dispositivo vinculado a él se le asigna una IP privada. Las IP privadas son estáticas; es decir, la dirección IP no cambia con el tiempo. Por tanto acceder a cada uno de los dispositivos si se está en la misma red es relativamente sencillo.

En cambio, en la red global, Internet, la direcciones IP públicas tanto del router como del Smartphone son dinámicas por lo que su IP pública cambia con el tiempo.

La opción elegida es la primera opción: comunicación en una **red Local LAN** debido a que es sencillo la comunicación entre ellos. La opción de conectarlos mediante Internet sería muy útil pero tiene una gran complejidad. El Smartphone no puede acceder al router porque éste cambia de IP pública cada poco tiempo. El Smartphone seguirá enviando a la misma dirección creyendo que el router recibe sus mensajes. Cuando el router cambie su dirección pública cambiará a otra IP desconocida de modo que no pueda conectarse el Smartphone con el router y viceversa. Por último cabe destacar que el Smartphone también tiene una dirección pública dinámica.

Este problema podría ser resuelto mediante soluciones como DynDNS o No-IP, sistemas que permiten emplear direcciones URL virtuales asociadas a IP dinámicas, de manera que el teléfono se conecta a esa dirección URL y recibe la IP que tiene el router en ese momento. En la actualidad la solución DynDNS es de pago, lo que supone un hándicap de cara a su uso extensivo en viviendas.

4. IMPLEMENTACIÓN DEL PROYECTO

4.1. Sensores

Para este proyecto se usan diferentes sensores. En los siguientes apartados se detallarán con mayor profundidad cada uno de ellos.

4.1.1. Sensores De Corriente

4.1.1.1. Introducción

El sensor de corriente utilizado es una pinza amperiométrica de la casa Efergy. El funcionamiento de la pinza se basa en la medida indirecta de la corriente circulante por un conductor (véase Imagen 15), gracias a un campo magnético. Recibe el nombre de pinza porque consta de un sensor, en forma de pinza, que se abre y abraza el cable cuya corriente queremos medir.

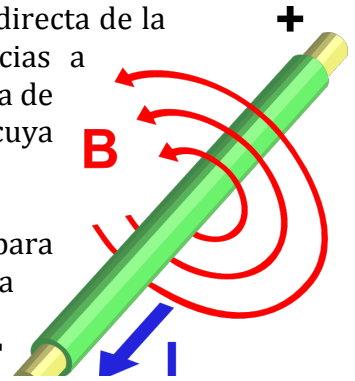


Imagen 14. Campo magnético (B) creado por una intensidad I

Este método evita la necesidad de abrir el circuito para efectuar la medida, así como las caídas de tensión que podría producir un instrumento clásico. Es mucho más seguro este método para medir la intensidad debido a que si no el operario tendría que estar en contacto con el cobre donde se estaría circulando corriente.

Imagen 15. Pinza amperiométrica

4.1.1.2. Cálculo de consumo

Para este proyecto, se quiere medir las corrientes que se consumen en una vivienda media en España. Según el Ministerio de Industria, la **potencia media contratada** en una vivienda Española es de **4,4KW**, lo que equivale a 4400 W.

La ecuación de potencia eléctrica viene dada por:

$$P\ (W) = I\ (A) \times V\ (V) \times FP$$

donde:
P = Potencia en watt
I = Intensidad en amperios
V = Tensión en voltios. 220V en viviendas
FP = Factor de potencia. 0,9 en viviendas

$$4400\ W = I\ (A) \times 230\ V \times 0{,}9$$

Lo que da una **intensidad máxima de fase de 21,25A**. Esta es la intensidad máxima que podrá circular en la vivienda. Si la intensidad fuera mayor debería saltar la Interruptor de Control de Potencia (ICP) colocado por la compañía eléctrica. Por tanto el siguiente paso será calibrar el sensor. Ha de medirse la tensión de salida que da el sensor para unas intensidades dadas.

4.1.1.3. Pruebas de calibración

- HORNO DE FUNDIDO

Primeramente se probó con un horno que funciona como un autotransformador. El primario se conecta a la red, es decir, a 230V eficaces. El secundario está conectado a una carga (el horno, de impedancia 89 Ω). Debido al autotransformador, se crea una tensión en el secundario, de modo que, como el secundario está conectado a la carga de 89 Ω, se crea una intensidad que es la que se va a medir. La tensión del secundario se puede controlar mediante el autotransformador de modo que se pueden crear diferentes intensidades en la carga (el horno).

Imagen 16. Esquema eléctrico del hornillo

La intensidad en el secundario se mide gracias a un multímetro digital. La salida del sensor será observada y analizada en un osciloscopio. La imagen del montaje que se hizo para estas pruebas en el laboratorio es la imagen 18.

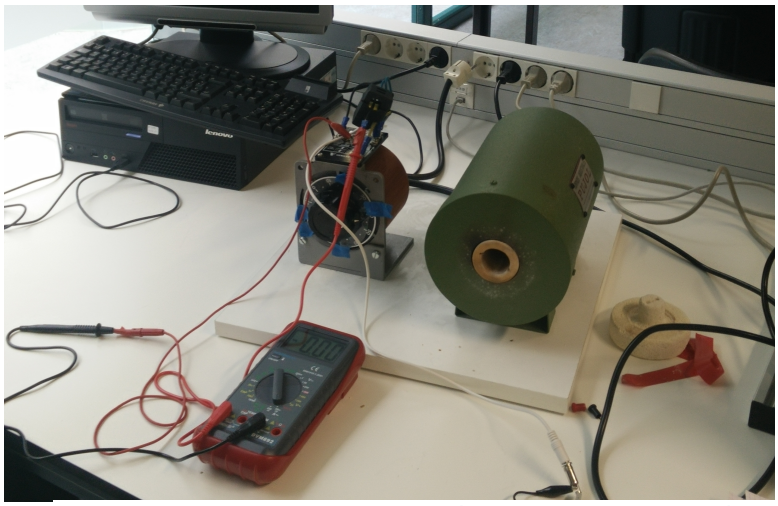


Imagen 17. Montaje para la calibración del sensor amperiométrico

La intensidad máxima que se puede obtener en el horno es:

$$I = \frac{V}{R} \rightarrow I = \frac{230\ V}{89\ \Omega} = 2{,}58\ A$$

por lo que, según lo mencionado en el apartado anterior de *cálculos de consumo* no es suficiente. Si se quiere llegar a los 21,25 A no vale con saber cómo funciona el sensor en 2,58A. Por tanto se necesita otra prueba que mida todo el rango de posibles valores de corriente de entrada.

- BANCO DE RESISTENCIAS

Estas pruebas se hicieron con un banco de resistencias. En ellas se pueden conectar diferentes resistencias en paralelo. Las resistencias que incluye el banco de resistencias son: 55 Ω, 110 Ω, 220 Ω y 440 Ω. Las intensidades teóricas se muestran en la siguiente tabla. (Los valores son en RMS)

Resistencias en paralelo				Resis equiv (Ω)	I Teoric (A)
440,00	-	-	-	440,00	0,52
220,00	-	-	-	220,00	1,05
220,00	440,00	-	-	146,67	1,57
110,00	-	-	-	110,00	2,10
110,00	440,00	-	-	88,00	2,62
110,00	220,00	-	-	73,33	3,15
110,00	220,00	440,00	-	62,86	3,67
55,00	-	-	-	55,00	4,20
55,00	440,00	-	-	48,89	4,72
55,00	220,00	-	-	44,00	5,25
55,00	220,00	440,00	-	40,00	5,77
55,00	110,00	-	-	36,67	6,30
55,00	110,00	440,00	-	33,85	6,82
55,00	110,00	220,00	440,00	29,33	7,87
55,00	110,00	220,00	-	31,43	7,35
55,00	55,00	-	-	27,50	8,40
55,00	55,00	110,00	-	22,00	10,50
55,00	55,00	55,00	-	18,33	12,60
55,00	55,00	55,00	55,00	13,75	16,80
110,00	27,50	55,00	55,00	12,22	18,90

Tabla 1. Intensidad Teórica obtenida del banco de resistencias

4.1.1.4. Resultados

- HORNO

I rms (A)	V (mv) pico-pico
0	20
0,3	141
0,4	185
0,5	225
0,6	265
0,7	310
0,8	350
0,9	394
1	438
1,1	482
1,2	531
1,3	575
1,4	619
1,5	659
1,6	704
1,7	744
1,8	800
1,9	840
2	880
2,1	910
2,2	930
2,3	960
2,36	965

Tabla 2. Valores obtenidos del sensor amperiometrico

Poniendo estos datos en una gráfica y obteniendo la recta de linealidad se obtiene:

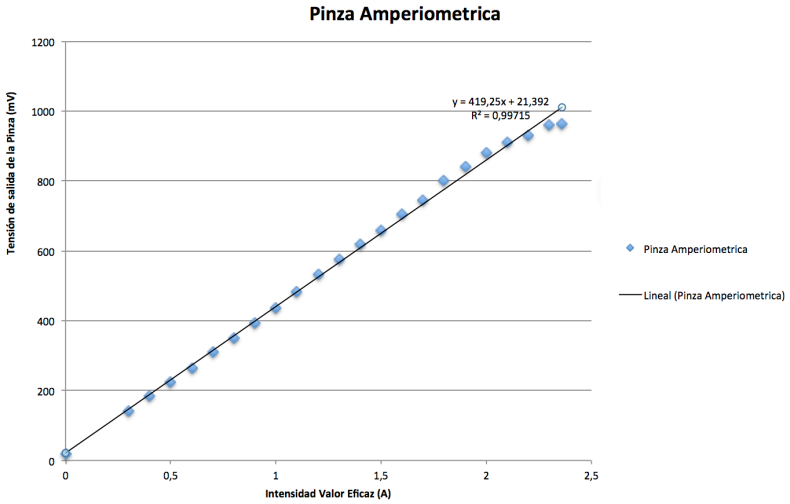


Imagen 18. Gráfica pinza amperiométrica Tensión de salida VS Intensidad

Se puede observar que la ecuación que se obtiene es **y= 419,25x + 21,392** y una **R^2=0,99715**. Por tanto, se puede afirmar que en el rango observado la relación se puede decir que es lineal debido a que la R^2=0,99715 y se rige por la ecuación antes descrita.

$$\boldsymbol{Tensión\ (mV) = 419.25 \times Intensidad\ (A) + 21.392}$$

- BANCO DE RESISTENCIAS

Resis equiv (Ω)	I Teórica (A)	I Real (A)	V salida (mV)
440,00	0,52	**0,45**	**69,7**
220,00	1,05	**0,99**	**150**
146,67	1,57	**1,45**	**218**
110,00	2,10	**2,01**	**301**
88,00	2,62	**2,48**	**363**
73,33	3,15	**3,02**	**420**
62,86	3,67	**3,48**	**457**
55,00	4,20	**4,05**	**492**
48,89	4,72	**4,51**	**516**
44,00	5,25	**5,05**	**539**
40,00	5,77	**5,52**	**556**
36,67	6,30	**6,06**	**574**
33,85	6,82	**6,55**	**588**
29,33	7,87	**7,55**	**615**
31,43	7,35	**7,10**	**603**
27,50	8,40	**8,12**	**629**
22,00	10,50	**10,11**	**676**
18,33	12,60	**12,12**	**709**
13,75	16,80	**16,08**	**770**
12,22	18,90	**18,16**	**796**

Tabla 3. Pruebas banco de resistencias

Los datos obtenidos se aproximan a los supuestos teóricamente. Poniéndolo en una gráfica se obtiene lo siguiente:

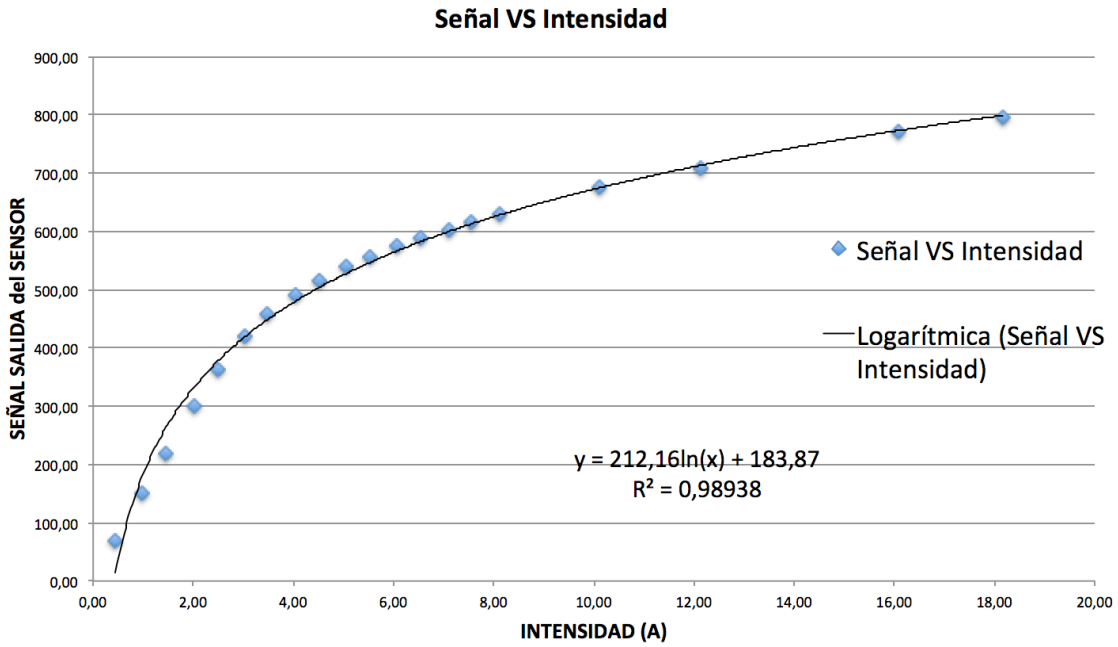


Imagen 19. Gráfica pinza amperiométrica Tensión de salida VS Intensidad

En este caso se tiene un rango mucho más amplio que llega hasta los 18 A. Se aprecia que en el rango de 0 - 3A la curva no se ajusta con demasiada fidelidad a lo que se supone debería. Por tanto, si la intensidad que se tiene es menor a 3 A se tendrá en cuenta la ecuación de la imagen 19 debido a que en ese rango es más preciso. Para intensidades mayores a 3 A la ecuación de la imagen 20 será la que se use. La ecuación es la siguiente:

$$\boldsymbol{Tensión\ eficaz\ (mV) = 212.16 \times ln(\ Intensidad(A)\ ) + 183.87}$$

$$\boldsymbol{R^2 = 0{,}98938}$$

4.1.1.5. Circuito de acondicionamiento de la señal

Según los cálculos anteriores se supone que a 21,25 A se obtendrá una señal de salida de;

$$\boldsymbol{Tensión\ eficaz(mV) = 212.16 \times ln(\ 21,25\ ) + 183.87 = 832.30\ mV}$$

Esta tensión es la tensión eficaz. La tensión que realmente se obtiene del sensor es la tensión pico-pico. Por tanto, la tensión pico-pico se obtiene fácilmente:

$$\boldsymbol{Tensión\ pico - pico = \left(832.30\ mV \times \sqrt{2}\ \right) \times 2 = 2354.12\ mV = 2.35412\ V}$$

La señal del sensor irá a convertidor analógico-digital (CAD) del Arduino para que este pueda procesarla. El CAD mencionado soporta un máximo de entrada de 5V por lo que lo óptimo sería que la tensión entrante al Arduino a la intensidad de 21.25 A fuera 5V para así tener mejor precisión. Por tanto, la señal necesita una ganancia;

$$\boldsymbol{Ganancia = \frac{5}{2.354} = 2.124}$$

Por otra parte el CAD del Arduino no soporta tensiones negativas por lo que si solo se le aplicara ganancia a la tensión esta tendría parte positiva y negativa debido a que la señal sinusoidal estaría centrada en 0. Por eso, se debe añadir un **offset de 2,5 V** de modo que a la máxima intensidad la señal oscile entre 0V y 5V.

Para realizar todo lo indicado arriba se utilizará un amplificador de instrumentación INA118. El amplificador de instrumentación está diseñado para tener una alta impedancia de entrada y un alto rechazo al modo común (CMRR). La operación que realiza es la resta de sus dos entradas multiplicada por un factor de ganancia. Una entrada será la tensión de salida del sensor y la otra la tierra del Arduino.

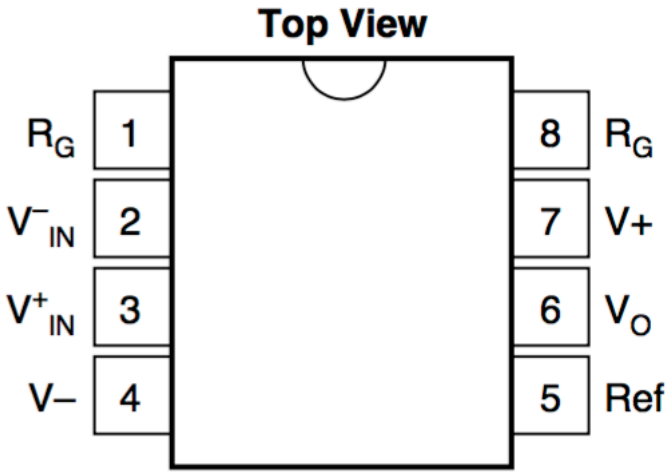


Imagen 20. Vista superior del amplificador INA 118P

Según *Texas Instrument*, fabricante del amplificador, la ganancia viene dada por: (Hojas de especificaciones adjuntada en el apartado *Anexo*)

$$G = 1 + \frac{50\ K\Omega}{R_g}$$

En consecuencia, si se quiere una ganancia de aproximadamente 2 (debería ser de 2.12 pero se deja un margen por seguridad) se debe elegir una R_g de 50 KΩ. Como las resistencias que existen en el mercado están normalizadas y no pueden tener cualquier valor que uno desee, se elige una resistencia de 47 KΩ y un potenciómetro de 10 KΩ en serie para poder hacer un mejor ajuste. Dichas resistencias tendrán que ir conectadas a las patillas 1 y 8 del amplificador.

Las patillas 2 y 3 del INA 118P son para las dos señales de entrada mientras que las patillas 4 y 7 están reservadas para las tensiones de alimentación. La patilla 6 es la patilla donde sale la tensión deseada y por último la patilla 5 es el terminal de referencia, que usaremos para añadir un offset de 2.5 V a la salida.

Para la inclusión de la tensión de offset, se utilizará un diodo Zener. Este es un diodo preparado para trabajar en la zona de ruptura.

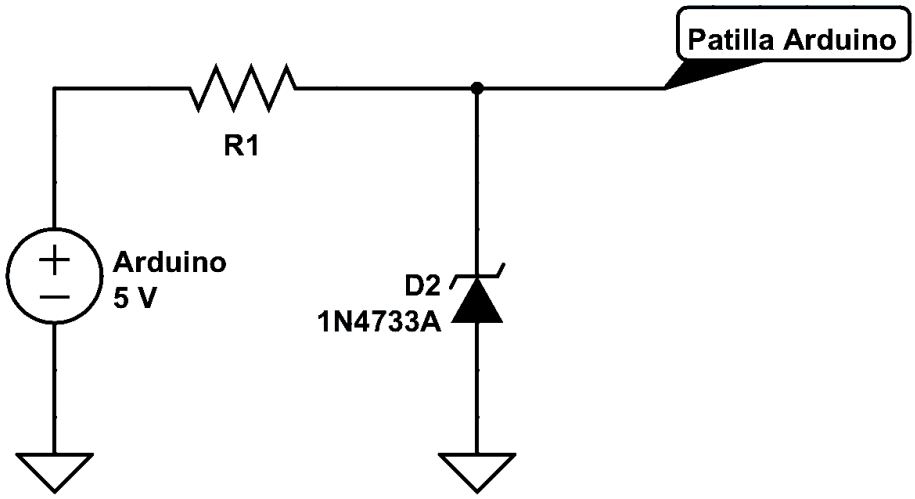


Imagen 21. Zener trabajando en la zona de ruptura

Como se aprecia en la Imagen 22, el esquema es muy sencillo. EL Arduino proporciona 5V de modo que debido a la resistencia R_1 cae tensión y el zener conduce esa intensidad en inversa. Si se le suministra tensión eléctrica positiva de cátodo a negativa en el ánodo (polarización inversa), el diodo mantendrá una tensión constante llamada Tensión Zener. Por tanto la tensión que hay en la patilla Arduino es de 2.5V debido a la tensión Zener. La R_1 tiene que ser un valor suficientemente grande como para que el zener trabaje polarizado en inversa y que no esté en corte. Se ha probado con una R_1 de 1KΩ y ha dado buenos resultados. Por tanto, la **R_1 escogida es de 1KΩ.**

4.1.2. Sensor de Temperatura

4.1.2.1. Introducción

En el mercado actual se pueden encontrar diferentes tipos de sensores de temperatura. Los sensores de temperatura son dispositivos que transforman los cambios de temperatura en cambios de señales eléctricas que son procesados. Principalmente hay tres tipos de sensores de temperatura: Termistor, RTD y los termopares. Para este proyecto se usará un termistor.

Los termistores son sensores de temperatura resistivos donde una resistencia cambia su resistencia de acuerdo con las variaciones de temperatura. Existen dos tipos de termistor, aquellos cuya resistencia aumenta en función de la temperatura, llamados PTC y aquellos cuya resistencia disminuye conforme aumenta la temperatura, los NTC. Éste segundo tipo es el que se usará para este proyecto.

En lo referente al proyecto y al sensor de temperatura se desea medir la temperatura de la vivienda. Por tanto, el rango que se desea medir es de 10ºC hasta los 40ºC.

4.1.2.2. Cálculos de Temperatura

La principal característica de los NTC es que son muy no lineales pero si muy sensibles a pequeños cambios de temperatura. La resistencia del sensor viene dadas por la siguiente ecuación:

$$R_t = R_0 \times e^{\frac{\beta}{T} - \frac{\beta}{T_0}}$$

donde;

- R_t es la resistencia del termistor
- R_0 es la resistencia a T=25ºC
- T_0 es la temperatura de 25ºC
- β es una constante definida por el fabricante

Debido a su respuesta exponencial hay que linealizarlo. Hay varias formas de linealizar, una de ellas es añadiendo resistencias. En este caso se pondrá una resistencia en paralelo con el NTC y otras en serie con estas éstas. En la imagen 22 se puede entender mejor el modo usado para linealizarlo.

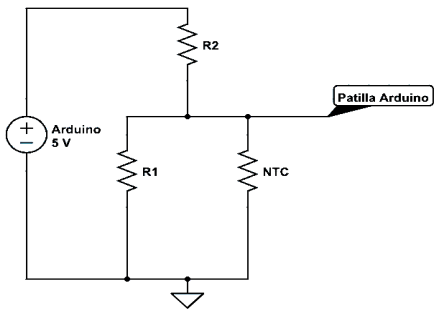


Imagen 22. Esquema electrónico linealización NTC

Ahora se intentará hallar el valor de R_1 y R_2 para que la señal de salida por la patilla del Arduino tenga una dependencia lineal con la temperatura. A bajas temperaturas se tendrá una mayor tensión de salida y viceversa. A una temperatura T_{min} se obtendrá una tensión V_{max} y a una temperatura T_{max} se tendrá una tensión V_{min}. Por experimentación sale que el máximo valor que se puede obtener de salida en la patilla del Arduino es de V_{max}=3.4V y V_{min}=2.1V a unas temperaturas de T_{max}=40ºC y T_{min}=10ºC, respectivamente.

Teórico						
R1	**R2**	**Rt**	**T**	**Beta**	**R eq**	**V out**
6591,93	702,86	1931,12	10,00	3700,00	1493,57	3,40
6591,93	702,86	1844,23	11,00	3700,00	1441,06	3,36
6591,93	702,86	1761,82	12,00	3700,00	1390,25	3,32
6591,93	702,86	1683,64	13,00	3700,00	1341,11	3,28
6591,93	702,86	1609,43	14,00	3700,00	1293,59	3,24
6591,93	702,86	1538,97	15,00	3700,00	1247,68	3,20
6591,93	702,86	1472,06	16,00	3700,00	1203,34	3,16
6591,93	702,86	1408,48	17,00	3700,00	1160,52	3,11
6591,93	702,86	1348,06	18,00	3700,00	1119,19	3,07
6591,93	702,86	1290,62	19,00	3700,00	1079,31	3,03
6591,93	702,86	1236,00	20,00	3700,00	1040,84	2,98
6591,93	702,86	1184,03	21,00	3700,00	1003,74	2,94
6591,93	702,86	1134,58	22,00	3700,00	967,98	2,90
6591,93	702,86	1087,51	23,00	3700,00	933,51	2,85
6591,93	702,86	1042,69	24,00	3700,00	900,29	2,81
6591,93	702,86	1000,00	25,00	3700,00	868,28	2,76
6591,93	702,86	959,32	26,00	3700,00	837,45	2,72
6591,93	702,86	920,56	27,00	3700,00	807,76	2,67
6591,93	702,86	883,60	28,00	3700,00	779,16	2,63
6591,93	702,86	848,36	29,00	3700,00	751,63	2,58
6591,93	702,86	814,74	30,00	3700,00	725,12	2,54
6591,93	702,86	782,66	31,00	3700,00	699,60	2,49
6591,93	702,86	752,04	32,00	3700,00	675,03	2,45
6591,93	702,86	722,81	33,00	3700,00	651,39	2,40
6591,93	702,86	694,90	34,00	3700,00	628,63	2,36
6591,93	702,86	668,23	35,00	3700,00	606,73	2,32
6591,93	702,86	642,75	36,00	3700,00	585,65	2,27
6591,93	702,86	618,40	37,00	3700,00	565,36	2,23
6591,93	702,86	595,11	38,00	3700,00	545,84	2,19
6591,93	702,86	572,85	39,00	3700,00	527,05	2,14
6591,93	702,86	551,55	40,00	3700,00	508,97	2,10

Tabla 4. Valores teóricos del circuito linealizador

4.1.2.3. Pruebas

Primeramente se va a comprobar que el sensor trabaja según como se ha descrito antes. Para eso se tomarán medidas de la resistencia del sensor a las temperaturas que se desean alcanzar con el sensor cuando esté operando, 10ºC-40ªC.

Temperatura medida	Resistencia ohm Practica	Resistencia Teórica
40	555,00	551,55
39	583,00	572,85
38	605,00	595,11
37	630,00	618,40
36	656,00	642,75
35	677,00	668,23
34	707,00	694,90
33	737,00	722,81
32	765,00	752,04
31	797,00	782,66
30	830,00	814,74
29	865,00	848,36
28	895,00	883,60
27	931,00	920,56
26	969,00	959,32
25	1005,00	1000,00
24	1047,00	1042,69
23	1092,00	1087,51
22	1140,00	1134,58
21	1192,00	1184,03
20	1241,00	1236,00
19	1302,00	1290,62
18	1349,00	1348,06
17	1402,00	1408,48
16	1467,00	1472,06
15	1533,00	1538,97
14	1612,00	1609,43
13	1670,00	1683,64
12	1746,00	1761,82
11	1821,00	1844,23
10	1900,00	1931,12

Tabla 5. Sensado NTC

Aunque exista un pequeño error se puede concluir con que el termistor se ajusta bien a la ecuación previamente descrita.

Una vez se sabe el valor de las resistencias que ayudan a linealizar el sensor de temperatura, se montó el circuito tal y como se demuestra en la *Imagen 22*. Las resistencias R_1 y R_2 tienen que tener valores normalizados, por lo que se eligen resistencias de valor:

- R_1= 6K8 Ω
- R_2= 680 Ω

Una vez escogidas las resistencias normalizadas se dispone a hacer las pruebas oportunas. El siguiente paso es observar la salida de tensión que da el circuito implementado para diferentes temperaturas.

R_t practico	R_1	R_2	R_{Eq}	V_{out}
1900,00	6800,00	680,00	1485,06	3,42
1821,00	6800,00	680,00	1436,35	3,39
1746,00	6800,00	680,00	1389,28	3,36
1670,00	6800,00	680,00	1340,73	3,32
1612,00	6800,00	680,00	1303,09	3,30
1533,00	6800,00	680,00	1250,98	3,24
1467,00	6800,00	680,00	1206,68	3,20
1402,00	6800,00	680,00	1162,35	3,16
1349,00	6800,00	680,00	1125,68	3,13
1302,00	6800,00	680,00	1092,77	3,11
1241,00	6800,00	680,00	1049,47	3,05
1192,00	6800,00	680,00	1014,21	3,01
1140,00	6800,00	680,00	976,32	2,96
1092,00	6800,00	680,00	940,90	2,92
1047,00	6800,00	680,00	907,30	2,87
1005,00	6800,00	680,00	875,59	2,83
969,00	6800,00	680,00	848,14	2,79
931,00	6800,00	680,00	818,89	2,75
895,00	6800,00	680,00	790,90	2,71
865,00	6800,00	680,00	767,38	2,68
830,00	6800,00	680,00	739,71	2,63
797,00	6800,00	680,00	713,39	2,59
765,00	6800,00	680,00	687,64	2,54
737,00	6800,00	680,00	664,93	2,50
707,00	6800,00	680,00	640,42	2,45
677,00	6800,00	680,00	615,70	2,39
656,00	6800,00	680,00	598,28	2,36
630,00	6800,00	680,00	576,58	2,31
605,00	6800,00	680,00	555,57	2,27
583,00	6800,00	680,00	536,96	2,22
555,00	6800,00	680,00	513,12	2,16

Tabla 6. Tabla sensado circuito acondicionador de Temperatura

4.1.2.4. Resultados

Todos estos datos son con los que va a trabajar el controlador del sistema, Arduino. Lo que esta vez se ha hecho es una vez se sabe el circuito acondicionador de la señal que se va a utilizar llevarlo a su lugar de trabajo, es decir, conectarlo con Arduino.

El CAD del Arduino tiene 10 bits por lo que tiene 1024 niveles diferentes que puede tomar. Estos niveles son de 0V hasta 5V. Por tanto, lo que se quiere es obtener para cada temperatura el nivel que hay.

La toma de resultados que se ha hecho ha sido de la siguiente manera: Se ha montado el circuito con las resistencias previamente descritas, y se sumerge el sensor NTC en agua caliente. Se mide la temperatura del agua y se observa el nivel del CAD que da el Arduino. Dejando un tiempo el agua se enfría y así se consigue diferentes temperaturas. Para tener un rango mayor se hace lo mismo con el agua fría y se deja que se caliente hasta Temperatura ambiente. Haciendo el experimento se obtiene la siguiente ecuación:

Temperatura	CAD
56	236
55	247
35	427
33	440
29	463
27	475
22	516
21	548
17	581
14	594

Tabla 7. Temperatura vs nivel del CAD

Poniéndolo en un gráfico se puede obtener su linealidad y la ecuación de la recta que lo representa.

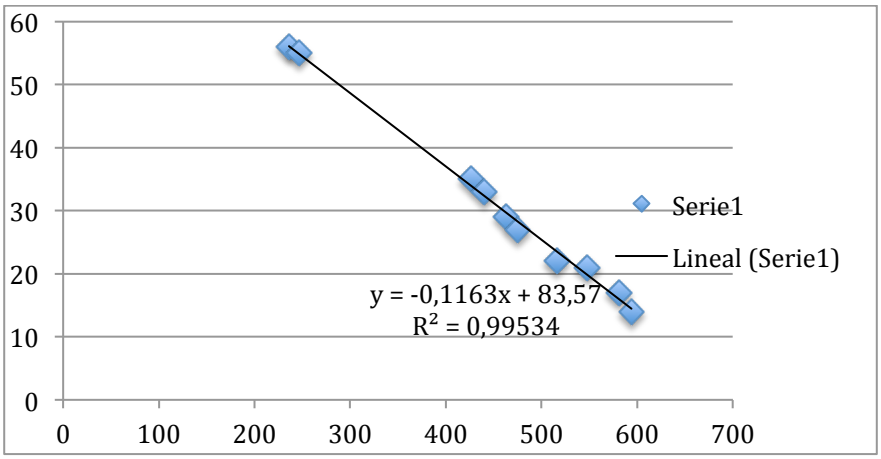


Imagen 23. Gráfica Temperatura vs nivel CAD Arduino

Por lo que, se obtiene la siguiente ecuación:

$$\boldsymbol{temperatura} = \mathbf{83.57} - \mathbf{0.11635} * \boldsymbol{z}$$

Donde;

- *Temperatura* es la temperatura medida por el sensor
- *z* es el nivel medido por el Arduino

La ecuación anterior será la que utilice el controlador Arduino para saber la temperatura que hace en cada momento sabiendo la entrada de tensión que tiene.

4.2. Arduino Shield

Una vez que se tienen los dos circuitos acondicionadores se procede a realizar el diseño del Arduino Shield. Un shield es una placa impresa que se pueden conectar en la parte superior de la placa Arduino para ampliar sus capacidades, pudiendo ser apilada una encima de la otra. En este caso se montará encima del Arduino Yún una placa donde irán los componentes de los circuitos.

4.2.1. DesignSpark

El diseño de la PCB o del Shield será hecho en el programa informático *DesignSpark. DesignSpark* es una herramienta de trabajo de diseño de placas electrónicas creada por *RS components*.

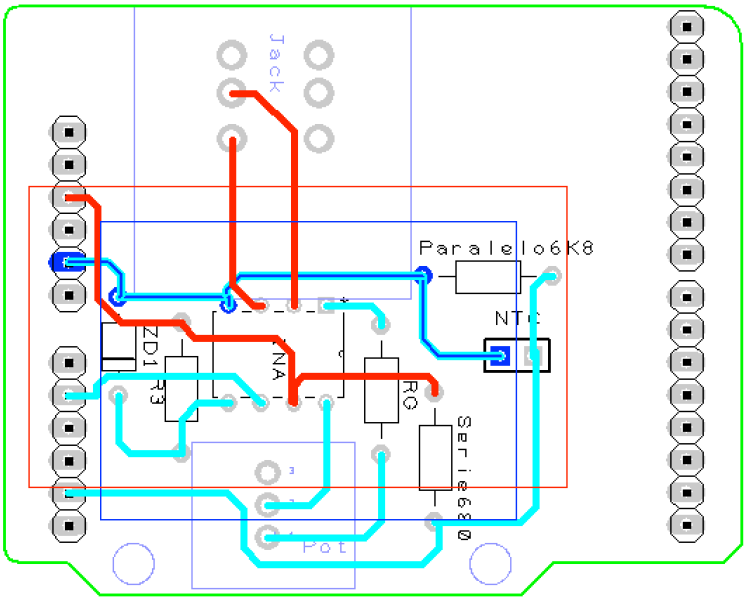


Imagen 24. Vista superior de la PCB

En la imagen superior se observa la disposición de los componentes dentro de la placa así como el enrutamiento entre ellos. Los conectores que están en ambos lados de la imagen son conectores con los que estará conectado al Arduino de modo que dé más estabilidad a la Placa o Shield.

El enrutamiento que está de color azul es la que está en la capa de abajo. Por el contrario las líneas que están en color rojo están en la capa de arriba. Por último hay que mencionar que el rectángulo rojo se llena de cobre el cual estará a 5V (en la capa superior). Por otro lado el rectángulo azul está a 0V en la capa inferior.

4.2.2. Vista 3D

La herramienta *DesignSpark* da la opción de ver el diseño de la placa a fabricar una vez terminada en 3D. Gracias a este software se obtienen las siguientes imágenes.

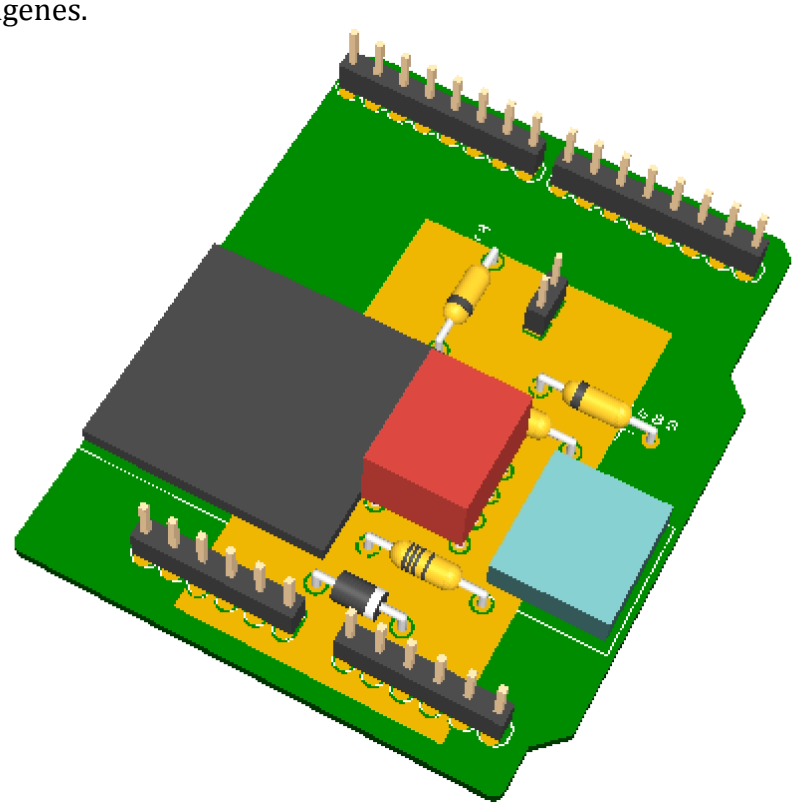


Imagen 25. Vista superior 3D de la placa

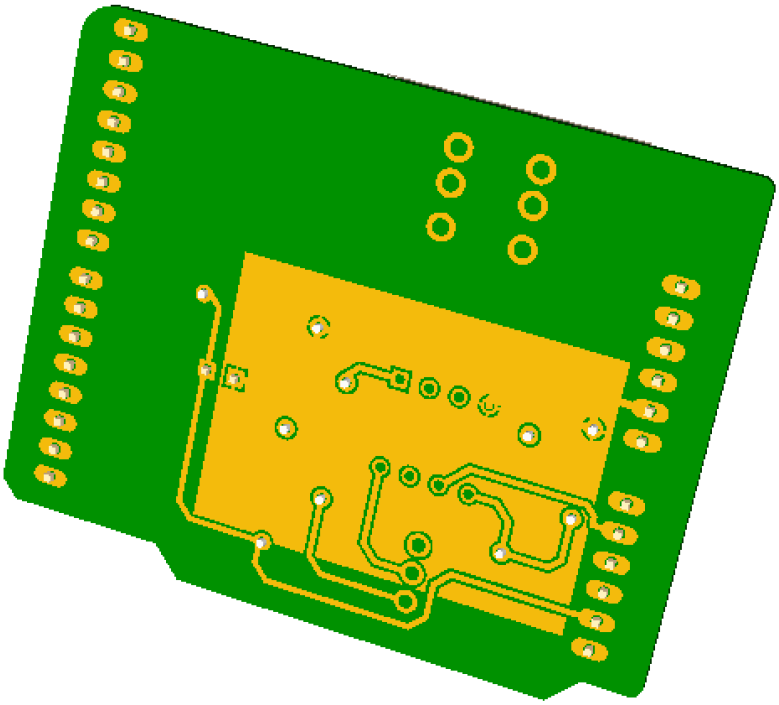


Imagen 26. Vista inferior 3D de la placa

4.2.3. Diseño real

Una vez diseñada la placa en *DesignSpark* se mandan los archivos necesarios a fábrica para que ésta pueda ser construida. Una vez la placa se fabrica el siguiente paso es **soldar** los componentes.

Se tiene por un lado la placa con los circuitos necesarios y por otro lado los componentes. Se suelda con cuidado y teniendo en cuenta el diseño antes establecido de modo que quede lo más parecido posible a la *imagen 24*. Se obtiene el siguiente Shield:

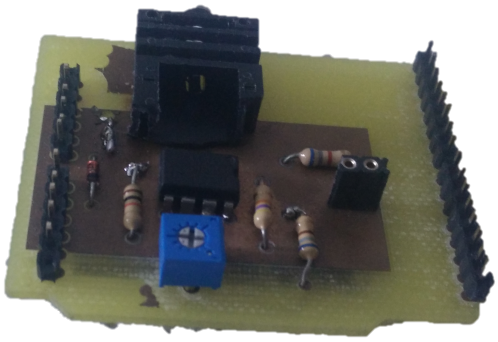


Imagen 27. Imagen del Arduino Shield

Se observa que a ambos lados hay conectores para que cuando se acople al Arduino quede estable y en caso de en un futuro querer hacer una instalación mayor o más grande tener los pines accesibles. En cada sitio ha sido situado las resistencias, diodos o componentes que deben estar. Solo falta la resistencia NTC. Esta, en vez de ir soldada directamente a la placa se ha puesto dos conectores (las que están a un lado del plano que se ve en la imagen 26) de modo que dicha resistencia se puede situar en cualquier lado de la casa, sin necesidad de que esté conectada al shield.

4.2.4. Problemas/errores

A la hora de llevar a cabo la soldadura o de unir los componentes con las placas surgieron diversos errores o problemas de diseño que obligó a hacer ciertas reparaciones en ella.

4.2.4.1. Conector Jack

Una de ellas fue que el conector Jack era más grande de lo que se pensaba. Por tanto, al intentar soldarlo el conector Jack pegaba al amplificador de instrumentación. La solución que se llevó a cabo fue cortar la punta del conector Jack ya que era solo plástico y no interfería en el funcionamiento del mismo. Así se pudo colocar.

4.2.4.2. Conexiones en la capa superior

Otro problema es que se diseñó que las patillas del conector Jack y el conector de 5V proveniente del Arduino para que se conectara en la capa superior. Esto fue imposible de llevarlo a la práctica ya que si se hiciera esto, los componentes no quedarían bien sujetos y no harían bien contacto. Es decir, se tendrían problemas de conexión. Por tanto, como se puede ver en la imagen se soldó a la capa inferior y las conexiones se hicieron con unos cables.

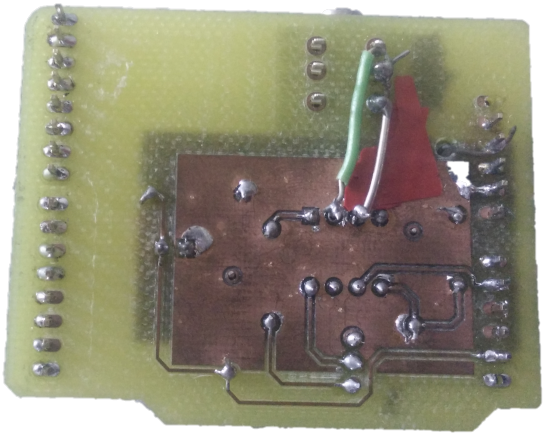


Imagen 28. Imagen de la capa inferior del Shield

4.3. DomoticApp

Como el propio título del documento destaca el proyecto trata sobre el control mediante un Smartphone de una vivienda es vital crear una buena plataforma con la que poder controlar la vivienda. Para ello se ha hecho una App desde la cual se podrá visualizar al mismo modo que también controlar las diferentes necesidades del cliente; en este caso, luces, temperatura y energía del hogar.

Es de vital importancia que la aplicación sea **organizada, sencilla y concisa**. Se trata que una persona que nunca ha interactuado con un Smartphone sea capaz de entender y de poder usarla en poco tiempo sin que le suponga demasiado esfuerzo llegar a entenderlo.

La aplicación ha sido realizada en la plataforma de Google *Android Studio* (para más información ver apartado *3.2.2. App Smartphone*). La App se ha dividido en diferentes *activities*. Un *activity* es cada una de las pantallas o vistas que forman una aplicación. A continuación se explicará las distintas *activities*, funcionalidades de la App así como la conexión entre ellas.

Cuando se descargar la App y se abre por primera vez, se pedirá un usuario y contraseña como modo de verificación. El usuario al ser un cliente se le facilitará el usuario y la contraseña de modo que pueda acceder a la App sin ningún problema. Así cada usuario podrá acceder fácilmente a su panel de control personalizado de la App. Por lo que para poder acceder a la App necesita ser cliente. Si por lo que sea el usuario olvida la contraseña se podrá modificar fácilmente gracias a la opción de resetear contraseña.

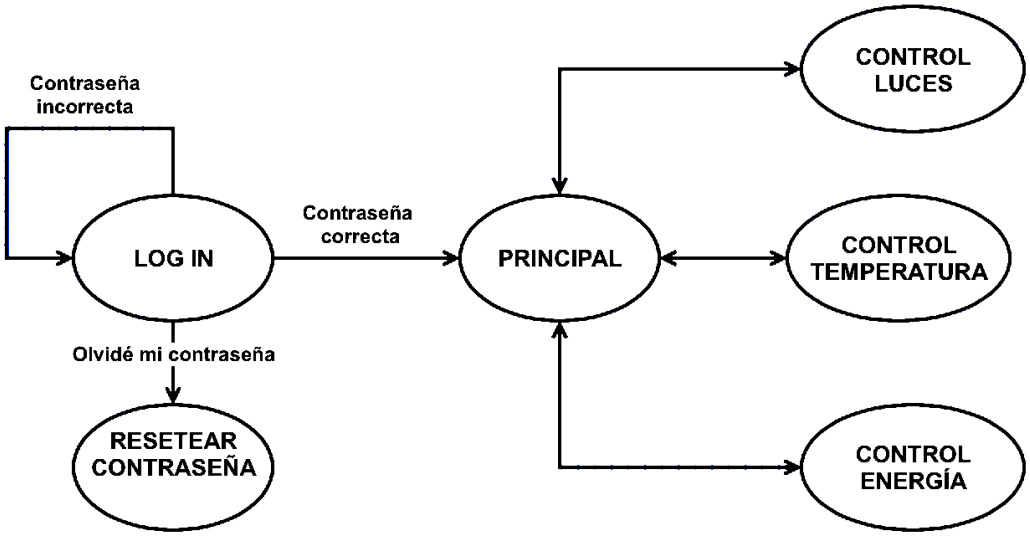


Imagen 29. Diagrama de flujo de la App

Una vez que el usuario accede al panel principal, donde en un vistazo se puede ver el estado de las luces, la temperatura y la energía, éste tiene la opción de ir al panel de control de la iluminación, panel de control de la temperatura o panel de control de la energía.

4.3.1. Log In

La Activity denominada Log in es la primera pantalla que aparece al abrir la aplicación.

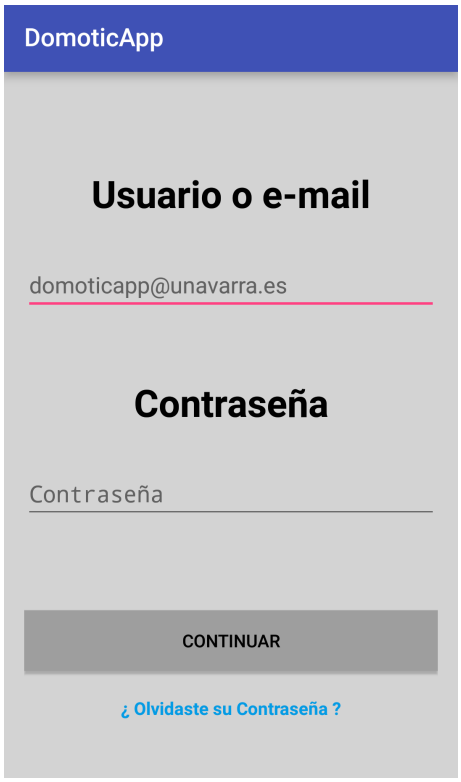


Imagen 30. Activity Log In

En esta se introduce el usuario facilitado y la contraseña y pulsando el botón llamado *CONTINUAR* para poder introducirse en el área personal donde se accederá al llamado Activity Principal. En caso de introducir erróneamente alguno de estos datos la App no dejará pasar a la siguiente Activity.

En el caso de no conocer u olvidar la contraseña pulsando en el texto "*Olvidaste su Contraseña"* se accederá al panel donde se podrá resetear la contraseña.

4.3.2. Resetear Contraseña

En este activity se da la opción de cambiar la contraseña. Si el usuario ha olvidado la contraseña o ha tenido algún problema con ella puede cambiar fácilmente en esta pantalla.

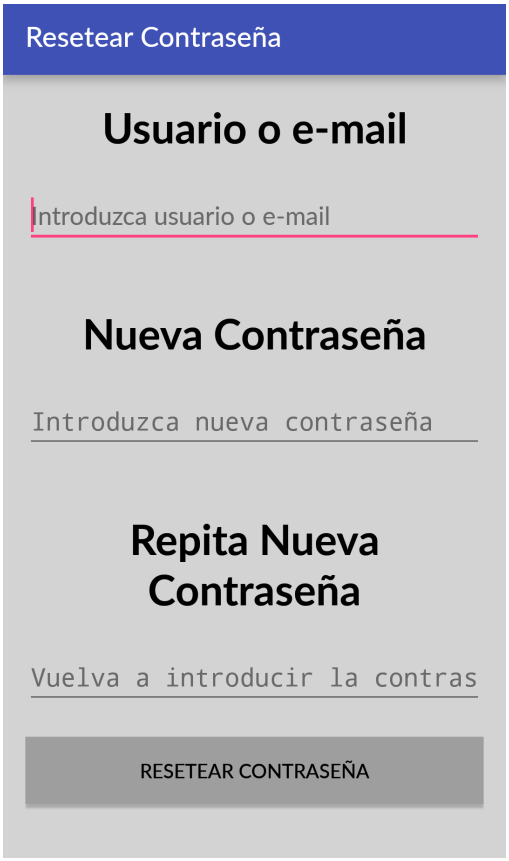


Imagen 30. Activity Resetear contraseña

Primero el usuario tiene que introducir que se le ha sido facilitado al usuario o el e-mail nada más contratar el proyecto. A continuación se pide que se escriba dos veces la misma contraseña para cerciorarse de que no ha habido ningún error tipográfico y la contraseña escrita es realmente la que el usuario quiera escribir. Una vez escrita correctamente la contraseña quedará reseteada. Si se escribe mal el usuario la aplicación avisará de que ha habido un error. También si las contraseñas escritas son diferentes avisará y se pedirá que se vuelvan a escribir para poder configurar bien la nueva contraseña.

4.3.3. Panel Principal

Una vez se ha introducido correctamente el usuario y la contraseña la aplicación abre el Activity principal. En él en un vistazo se puede ver todos los estados de as variables que se quieren controlar de la casa.

Imagen 31. Activity Principal

Por un lado se tienen tres botones los cuales llevan de una Activity a otra. Dichos botones son " *Control de iluminación*", "*Control de Temperatura*" y "*Control de Energía*". Debajo de el botón de *control de iluminación* se puede ver el estado de las luces. En este caso las luces que se pueden controlar son las de la cocina, Salón, baño y pasillo.

Por otro lado debajo del botón de control de temperatura se observa cuál es la temperatura actual de la vivienda así como la temperatura de consigna. La temperatura de consigna es la temperatura que el usuario desea que haya en su vivienda. Además dicha temperatura es controlable desde este Activity gracias a una barra situada debajo de la temperatura de consigna tal y como se ve en la imagen.

Por último se encuentra el botón de Control de energía.

4.3.4. Control de Iluminación

Si en el Activity principal se hace *click* en el botón de control de iluminación se accede al siguiente Activity.

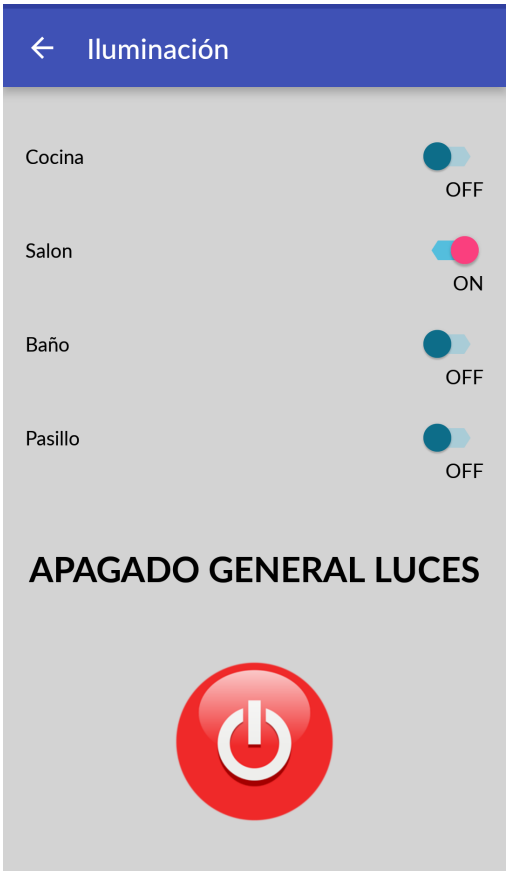


Imagen 32. Activity iluminación

Se puede apreciar que hay cuatro *switch*-es con los cuales se puede cambiar el estado de las variables. Además cuando cambia el estado de la variable se cambia el texto que está debajo de cada *switch* para ayudar a una mejor visualización de los estados al usuario.

Debajo de estos switch es se puede leer "APAGADO GENERAL LUCES" y debajo un botón rojo. En caso de que este botón sea pulsado, se apagarán automáticamente todas las luces, de modo que no haya que ir uno a uno apagándolos haciendo que se pierda menos tiempo.

Por último, en la parte izquierda de la cabecera hay una flecha. Esta flecha indica que el usuario, una vez hechos todos los cambios deseados puede volver al Activity principal para poder ver el estado de otras variables o para poder ir a otros activities.

4.3.5. Control de Temperatura

Si en el Activity principal se hace *click* en el botón de control de temperatura se accede al siguiente Activity.

Imagen 33. Activity Temperatura

En la parte superior de este Activity se muestra el texto *TEMPERATURA ACTUAL* y debajo de éste la temperatura que hace en el hogar, en este caso 19ºC. Debajo de esto se encuentra la temperatura de consigna con la temperatura que el usuario quiere. Debajo se encuentra un botón "+" que cuando se cliquea la temperatura de consigna sube 1ºC. En cambio si se pulsa el botón "-" azul, disminuye en 1ºC la temperatura de consigna. La temperatura mínima es 10ºC y la máxima 35ºC de modo que cuando la ésta sea menor o mayor, los botones - y + dejarán de funcionar, respectivamente.

Por último, y como en el apartado anterior, en la parte izquierda de la cabecera hay una flecha. Esta flecha indica que el usuario, una vez hechos todos los cambios deseados puede volver al Activity principal para poder ver el estado de otras variables o para poder ir a otros activities.

4.3.6. Control de la energía

Si en el Activity principal se hace *click* en el botón de control de energía se accede al siguiente Activity.

Imagen 34. Activity Control de energía

En esta Activity se muestra el consumo de potencias del mes en curso, así como el del mes anterior y el consumo máximo y mínimo hasta la fecha. Una vez se tienen las potencias consumidas es muy fácil estimar cómo afectará económicamente estos datos de consumo.

Por último, como en las Activities anteriores en la parte izquierda de la cabecera hay una flecha. Esta flecha indica que el usuario, una vez hechos todos los cambios deseados puede volver al Activity principal para poder ver el estado de otras variables o para poder ir a otros activities.

4.4. Arduino Yún

El controlador del sistema es el Arduino Yun. (información previa en 3.2.1. Controlador del sistema)

4.4.1. Arduino Yún: A fondo

El Arduino Yun será el controlador usado en este proyecto. El procesador del Arduino Yún es el procesador *ATmega32u4.* También el Arduino Yun tiene otro procesador *Atheros AR9331* el cual es compatible con una distribución Linux basada en OpenWrt.

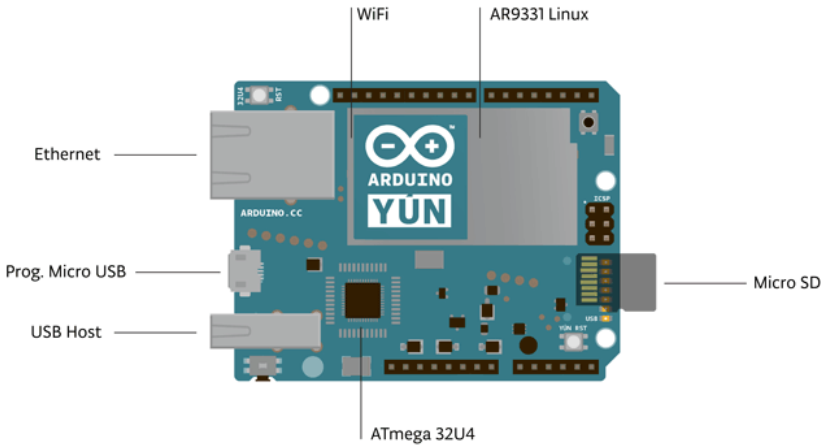


Imagen 2. Componentes Arduino Yún

La placa lleva incorporada un puerto Ethernet y soporte Wi-fi tal y como se puede observar en la imagen de arriba. También lleva una ranura para tarjeta micro SD , 20 pines de salidas o entradas digitales (de los cuales 7 se pueden utilizar como salidas PWM y 12 como entradas analógicas), un oscilador de cristal de 16 MHz, una conexión micro USB, una cabecera ICSP y un tres botones de reinicio; uno para el WLAN, otro del ATmega 32U4 y el última del Yún en general. A continuación se adjuntan dos tablas con las especificaciones de los dos procesadores de la placa.

Microcontrolador	ATmega32U4
Voltaje de trabajo	5V
Voltaje de entrada	5
Pines Digitales E/S	20
Pines PWM	7
Pines Analógicos	12
Corriente por pin	40 mA
Corriente por pin 3.3V	50 mA
Memoria Flash	32 KB
SRAM	2.5 KB
EEPROM	1 KB
Frecuencia de reloj	16 MHz

Tabla 8. Microcontrolador AVR Arduino

Procesador	Atheros AR9331
Arquitectura	MIPS @400MHz
Voltaje de trabajo	3.3V
Ethernet	IEEE 802.3 10/100Mbit/s
WiFi	IEEE 802.11b/g/n
USB Tipo-A	2.0 Host
Lector de tarjetas	Micro-SD
RAM	64 MB DDR2
Memoria Flash	16 MB
SRAM	2.5 KB
EEPROM	1 KB
Frecuencia de reloj	16 MHz

Tabla 9. Microcontrolador Linux

4.4.2. Sketch Bridge

La diferencia entre el Arduino Yun y los demás Arduinos es que el Yún es capaz de comunicarse con un pequeño sistema operativo llamado Linino gracias al procesador *AR9331* la cual puede funcionar como un ordenador en red de gran alcance con todas las facilidades de Arduino.

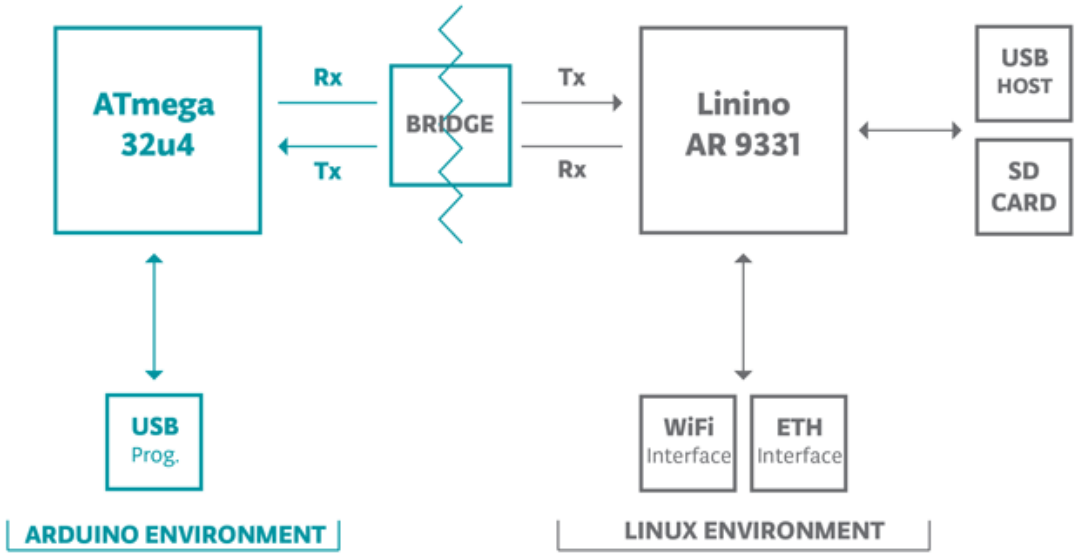


Imagen 363. Esquema Arduino Yún con Bridge

El Arduino Yún trabaja con el procesador *ATmega 32u4* en el entorno de Arduino. Pero gracias a una de las librerías que trae el IDE de Arduino (interfaz gráfica que ayuda a programar) el Arduino trabaja con el sistema operativo Linino. Dicha librería es la librería llamada *Bridge*. Gracias a Bridge se activa el pequeño sistema operativo en el microcontrolador AR9331 con la que se puede acceder a distintas funcionalidades tales como la tarjeta SD, Wifi, Ethernet, Host USB, etc., a la vez que se puede seguir trabajando en el Arduino normalmente.

4.4.3. Comunicación a través de WebServer

Cuando se accede a la librería Bridge el Arduino Yún hace de servidor web gracias al microcontrolador AR9331. Cualquiera dentro de la misma red en la que está conectado el Arduino al router puede acceder a dicha web. Abriendo un navegador y escribiendo en él una serie de URL definidas se puede obtener información sobre el Arduino.

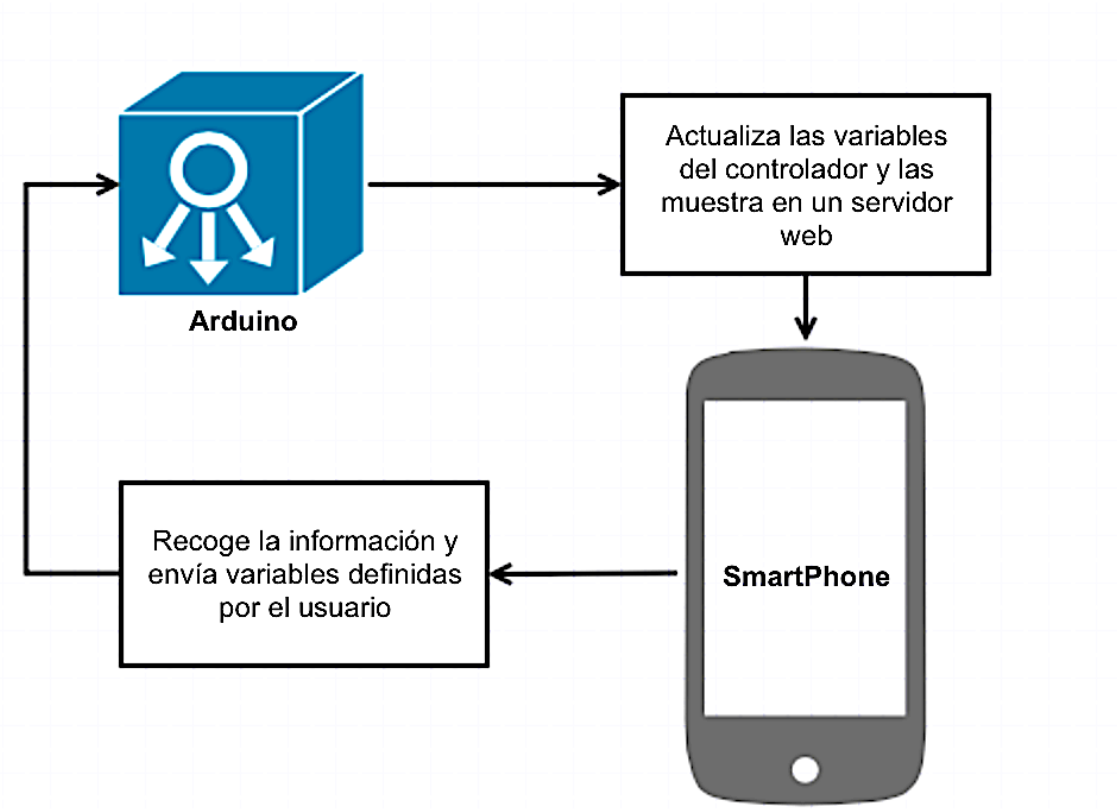


Imagen 37. Diagrama de comunicación

El Arduino teniendo en cuenta la URL que se ha escrito interpreta que cierta variable quiere ser cambiada.

```
"IPprivadaarduino/arduino/digital/13"     -> digitalRead(13)
"IPprivadaarduino/arduino/digital/13/1"   -> digitalWrite(13, HIGH)
"IPprivadaarduino/arduino/analog/2/123"   -> analogWrite(2, 123)
"IPprivadaarduino/arduino/analog/2"       -> analogRead(2)
"IPprivadaarduino/arduino/mode/13/input"  -> pinMode(13, INPUT)
"IPprivadaarduino/arduino/mode/13/output" -> pinMode(13, OUTPUT)
```

Imagen 384. Comandos URL

Escribiendo en cualquier navegador los comandos anteriores se puede acceder a cierta información de Arduino. Si la IP privada del Arduino fuera *192.168.1.46,* por ejemplo, se podríamos acceder a los diferentes comandos, como los mostrados a continuación:

- *192.168.1.46/arduino/digital/13*: Este comando da la información del estado del pin digital 13 (o la que se hubiera pedido en la URL). En el navegador aparecerán la siguiente información. "*Pin D13 set to 0*" en caso de que estuviera a 0 y "*Pin D13 set to 1"* en caso de que fuera 1.

- *192.168.1.46/arduino/digital/13/X (siendo X 0 o 1):* En este caso lo que se pretender es poner el pin digital 13 a X. La información que aparecerá en el navegador será la misma que en el anterior. "*Pin D13 set to 0"* en caso de que X sea 0 y "*Pin D13 set to 1"* en caso de que fuera 1.

- *192.168.1.46/arduino/analog/2/123:* este comando sirve para darle el valor de 123 al pin analógico 2. El CAD del arduino tiene 10 bits por lo que el rango de valores que se pueden establecer van desde 0 hasta 1023.

- *192.168.1.46/arduino/analog/2:* Con esta instrucción se pide la información del valor que está en el pin analógico 2 del arduino. Por lo que la información que se da en el navegador es, "*Pin A2 reads analog X". El valor X es el valor al que estará el pin Analógico 2 cuando se haga la solicitud.*

- *192.168.1.46/arduino/mode/13/X (donde X es input o output)*: De este modo se puede poner el pin 13 como entrada o como salida dependiendo de si la X es input o output.

Por tanto usando estos comandos de un modo adecuado se puede comunicar el módulo arduino con cualquier aparato dentro de la misma red, en ambos sentidos. Cualquier aparato puede poner a 1 o a 0 las salidas digitales, mirar sus estados o jugar con los pines analógicos.

5. COMUNICACIÓN ARDUINO - ANDROID

Como se ha comentado durante este documento, el proyecto llevado a cabo se desarrollará en el contexto de una red LAN. A continuación se detallará cómo se ha hecho la comunicación entre la App del Smartphone y el Arduino, siguiendo las instrucciones que se han comentado en el apartado anterior.

5.1. Ajustes Iniciales

Antes de empezar a hacer cualquier cosa es necesario configurar el Arduino. Hay que conectar el Arduino con red local para después intentar comunicarlo con el Smartphone dentro de dicha red.

5.1.1. Paso 1

El primer paso es conectar el Arduino a alimentación. Para eso gracias a un cable se conecta el Arduino mediante el micro USB con el puerto USB del ordenador. Una vez hecho esto, se encenderá el LED "ON" del Arduino, el LED L13 y el LED llamado USB. Si no se consigue esto, se puede resetear la configuración para volver al estado de fábrica. Para ello hay que apretar el botón llamado "WLAN RST" que está situado cerca del USB host (se muestra en la **imagen 13).**

5.1.2. Paso 2

Si todo se ha hecho correctamente debería salir una nueva red Wifi llamada Arduino Yún-XXXXXXXXXXXX (estos X son diferentes números y/o letras). El siguiente paso consiste en conectarse a esa red.

Imagen 39. Conexiones Wifi existentes

5.1.3. Paso 3

Una vez conectado el ordenador a la red Arduino Yún - XXXXXXXXXXXX se abre un navegador cualquiera y se accede a la siguiente URL: "Arduino.local". Esto lleva a la siguiente página donde se accede con una contraseña. Si no se ha cambiado ésta debería ser "*arduino*".

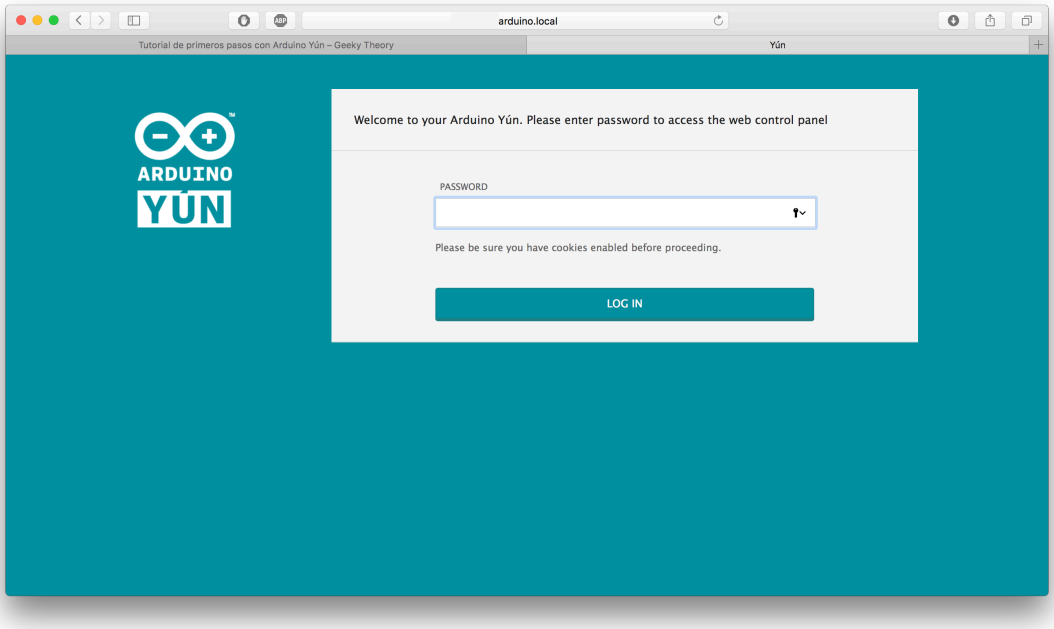


Imagen 40. Página que se accede a la configuración del Arduino

5.1.4. Paso 4

Cuando de introduce la contraseña correcta se accede a la siguiente página.

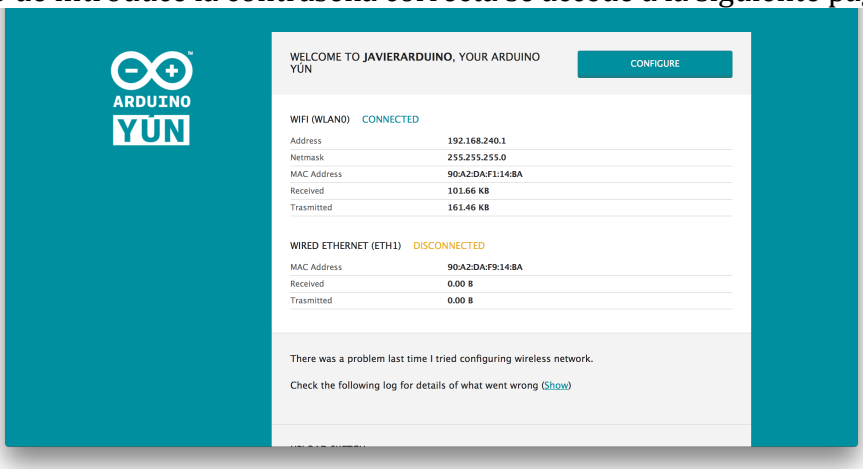


Imagen 41. Página configuración Arduino

En esta página se puede obtener información del Arduino. Una vez en esta página hay que darle a configuración para poder conectar el Arduino a la red.

5.1.5. Paso 5

Al darle a la pestaña superior que dice "CONFIGURE", se muestra la siguiente página.

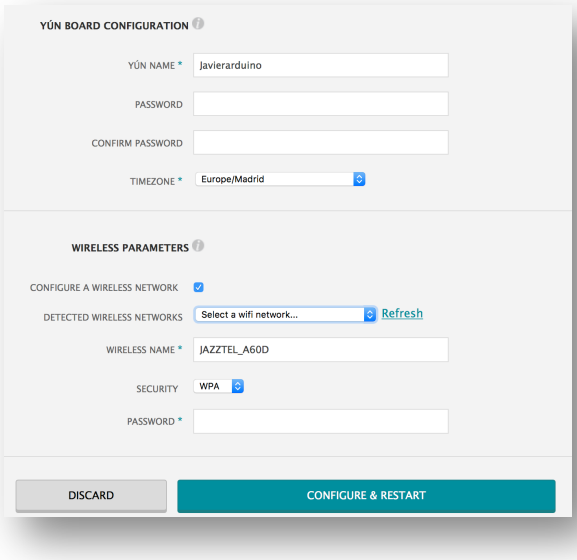


Imagen 42. Página configuración Arduino

Aquí clicka en la pestaña llamada *DETECTED WIRELESS NETWORK* y se selecciona sale un despegable con todas las redes wifi disponibles. Se elige la red wifi del hogar, en este caso "MOVISTAR_062C".

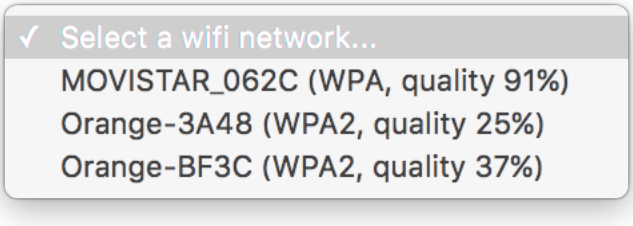


Imagen 43. Desplegable

A continuación ha de insertarse la contraseña del router en el hueco que está a la derecha de PASSWORD. Una vez introducida la contraseña se pulsa el botón CONFIGURE & RESTART y aparece lo siguiente:

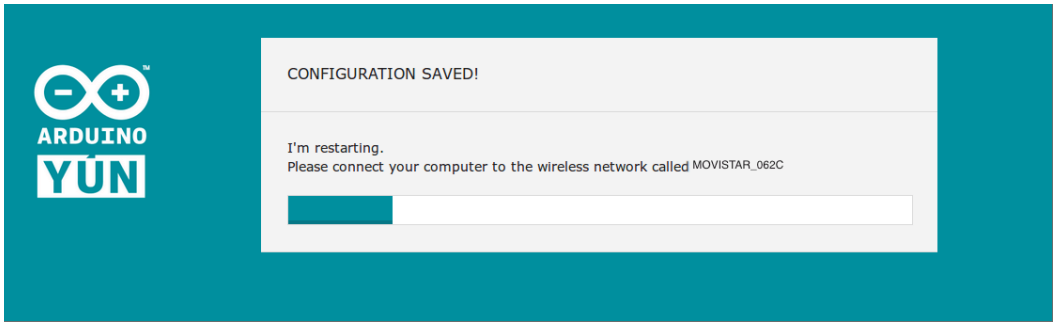


Imagen 44. Configuración Arduino

5.2. Comunicación del sistema

La comunicación es un aspecto básico y vital en este trabajo debido a que desde la App se desea enviar al Arduino diferentes ordenes u obtener en ella cierta información. Debido a esto, el sistema domótico debe estar en constante comunicación entre sí. A continuación se explicará cómo se ha llevado a cabo esta tarea.

Por un lado el Arduino gracias al sistema de acondicionamiento del Shield es capaz de obtener la temperatura actual de la vivienda en la patilla analógica 4 y la potencia instantánea consumida de la vivienda.

Por otro lado en la App se controlan y visualizan variables como el estado de las luces, la temperatura de consigna (la temperatura a la que el usuario desea que esté la vivienda). Por tanto, toda esta información ha de compartirse entre los dos equipos.

5.2.1. Luces

Cuando se desea encender una u otra bombilla se introduce una URL en el navegador tal y como se explicó en el apartado 4.3.3 Comunicación a través de WebServer. La luz de la cocina se activa en la patilla digital 2 del Arduino, la de la cocina en la 3, la del salón en la 4 y por último la del baño en la 5.

Hay que tener en cuenta que el estado de la iluminación del hogar puede modificarse por dos acciones. La primera de ellas por la App *DomoticApp.* Pero no es la única forma de cambiar el estado de las variables. Accionando el interruptor de casa también se puede apagar. Por tanto, la aplicación necesita saber si el estado de cada variable ha sufrido algún cambio que no haya sido causado por la App propia. Para ello y gracias a un Timer, la aplicación interroga cada 15 segundos al Arduino y recibe la información sobre estas variables. En caso de que haya algún cambio la App lo detecta y cambia el estado de la luz automáticamente.

5.2.2. Termostato

Para la Activity de termostato, el Arduino con el circuito de adecuación de la señal obtiene la temperatura actual en la patilla 4, por lo que interrogando al Arduino con una URL se obtiene fácilmente la temperatura existente en la vivienda. Una vez que la App obtiene la temperatura actual de la vivienda y debido a que la temperatura de consigna es otra variable más de la App no hay más que compararlas. Cuando la temperatura actual tenga un valor menor a la de consigna la caldera estará encendida, pero cuando la temperatura actual tenga un valor mayor a la de consigna la caldera estará apagada.

En este apartado se ha introducido una **histéresis** de 0.5ºC por eficiencia y para que la caldera no esté encendiéndose y apagándose cada poco periodo de tiempo.

5.2.3. Energía

La Activity Control de Energía, al igual que el de Termostato, se obtiene en la patilla 1 el valor de la intensidad que circula en la vivienda, con lo que fácilmente se obtiene la potencia consumida.

Una vez que se obtiene estos valores no hay más que guardar estos datos. El Arduino Yún ofrece la posibilidad de guardar datos de los consumos realizados los últimos meses gracias a la ranura de tarjeta SD.

6. PROYECTO FINAL

En este apartado mostrará el resultado final del proyecto.

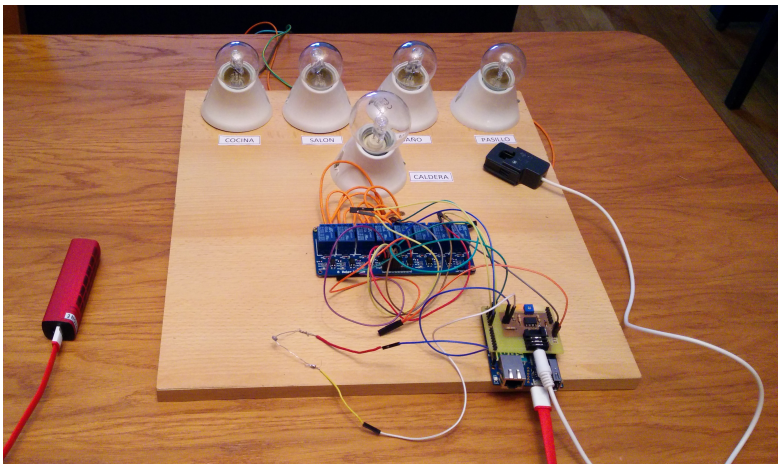


Imagen 45. TFG finalizado

Con objeto de mejorar la presentación se han atornillado 5 portalámparas a una base de madera. Sobre estas portalámparas se conectan las bombillas que simularán la iluminación del hogar y la caldera. Las 4 bombillas situadas en la parte trasera de la base, cada una con sus respectivos nombres al lado, simulan cada estancia de la casa. La bombilla que está más al centro y con unas dimensiones mayores simula la caldera.

Un enchufe alimenta por debajo de la base de madera a los relés los cuales son controlador por las portalámparas. Por último el Arduino está alimentado por una batería externa. Esto se hace así debido a que si la red fallará se podría perder información importante o la seguridad de la vivienda se podría ver afectada si el sistema se ocupase de ello.

Por último, en el siguiente enlace se muestra en un video el funcionamiento del TFG.

https://youtu.be/3r5QvUSYV3o

7. CONCLUSIONES

A continuación se expondrán los objetivos logrados durante la realización de este proyecto.

Se ha logrado la monitorización y control de la iluminación y temperatura de la vivienda basada en la plataforma Arduino.

Se ha logrado controlar de la energía consumida y aproximar las facturas mensuales así como guardar dichos datos.

Se ha logrado crear una interfaz sencilla con el usuario, ordenada e intuitiva.

Económicamente barato y extensible en el futuro. Esto se ha conseguido haciendo una PCB sencilla y gracias al Arduino Yún. La App es gratuita.

Actuación sobre diferentes elementos de la planta. Se actúa sobre relés que actúan sobre las bombillas. En caso de la temperatura el actuador actúa sobre la caldera activándola o desactivándola.

8. LINEAS FUTURAS

8.1. Correcciones

Teniendo en cuenta todo lo anteriormente expuesto en este apartado se realizará una segunda versión del proyecto reflejando todas las correcciones realizadas. Habría que rediseñar la tarjeta de modo que los problemas o errores mencionados en el apartado 4.2.4 Problemas /errores dejen de existir.

Por tanto por un lado habría que dejar más sitio para que el conector Jack no choque con ningún otro componente y pueda así encajar sin ningún problema en el PCB. Por otro lado habría que modificar el diseño del PCB para que las conexiones del conector Jack así como las conexiones a 5V estuvieran situadas en la capa inferior del PCB.

8.2. Mejoras

Este proyecto ha sido creado desde cero en apenas unos meses. Por eso en este apartado se plantean diferentes objetivos o líneas a seguir en el futuro para que el proyecto siga adelante. La domótica pretende aportar seguridad y confort al usuario por lo que no hay una sola manera de hacerlo. Para este proyecto se sugieren las siguientes mejoras o evoluciones del proyecto.

8.2.1. Seguridad anti-intrusión y comunicación

Uno de los mayores miedos de los usuarios es el miedo de que algún ladrón entre a robar cuando no haya nadie en la vivienda. Por eso es muy importante dotar al sistema domótico de seguridad frente a robos.

El usuario cuando salga de la vivienda podrá activas la seguridad anti intrusión. Esto activará unos sensores de presencia de modo que si alguien se acerca a la vivienda se podrán tomar diferentes medidas para evitar el hurto o la intrusión de la persona no deseada.

Por un lado el sistema detecta si alguien intenta entrar en la vivienda. Las acciones a tomar pueden ser diferentes. Una acciones que el sistema domótico encienda ciertas luces de la casa de modo que desde fuera de la vivienda parezca que el usuario está dentro de ella.

Si esto no ha conseguido hacer que el ajeno a la vivienda se vaya y consigue entrar en la vivienda, las puertas se cerrarían al mismo tiempo que se encenderían las luces y sonaría la sirena. Además el sistema avisaría directamente a las fuerzas policiales avisando del intento de hurto al mismo modo que se avisaría al usuario de la vivienda. Por último el usuario podría ver imágenes en directo del hogar gracias a diferentes cámaras situadas en él.

8.2.2. Alarmas Técnicas

El sistema domótico está equipado con diferentes sensores lo cual hace que se pueda detectar ciertas actividades que pueden ser peligrosas por el usuario.

Por ejemplo un detector de humo en diferentes zonas de la casa puede hacer que en caso de incendio el sistema llame a los bomberos si fuera necesario y haga que unos rociadores se activen echando agua en el sitio indicado con el fin de apagar dicho incendio. El detector de humo en la cocina podría hacer que pasado un umbral de humo activara automáticamente el extractor de humo para que el humo no perjudique la salud de los habitantes del hogar.

También podría ponerse sondas de agua. Estas sondas de agua colocadas en el lavadero y en los baños informan al sistema para actuar sobre las electroválvulas de corte de suministro de agua, impidiendo que si hay un escape de agua éste continúe y provoque mayores daños.

8.2.3. KeyFree

En este apartado se pretende aprovechar al máximo la accesibilidad a la vivienda gracias a la App del Smartphone. Se pretende buscar otra salida si al cliente se le han olvidado las llaves de casa.

Esta solución consiste en que en la App llamada *DomoticApp* tenga una Activity llamada llave. En ella aparecerá un código QR. Acercando dicho código a un lector de códigos que estará situado cerca de la puerta del hogar, éste lo leerá y se abrirá automáticamente la puerta.

8.2.4. Aumentar seguridad de la comunicación

A pesar de todos los aspectos que se puedan mejorar la seguridad de la comunicación es muy importante. Hay que evitar que terceras personas puedan acceder al sistema y permitan que caiga el sistema haciéndolo vulnerable a ataques desde fuera.

Por tanto para que la vivienda funcione bien y no se pueda conectar nadie que no sea el usuario, hay que dotarlo de mayor seguridad en cuanto a la comunicación.

8.2.5. Comunicación con IP dinámicas y fuera de red Local

Por último y una de las más importantes mejoras de cara al futuro es conseguir que la comunicación entre el dispositivo móvil y el Arduino se pueda producir en una red no local, a través de internet, donde la IP dinámica hace que la IP de cada dispositivo vaya cambiando.

9. BIBLIOGRAFIA Y REFERENCIAS

- [1] "*Sistema Distribuido vs. Sistema Centralizado, o Sistema Distribuido en un Sistema Centralizado*", Jacinto Fung.

Disponible en:
http://iscbunkerramo.blogspot.com.es/2011/11/sistema-distribuido-vs-sistema.html

- [2] "*Dispositivos de la vivienda domótica*", Ramón Jesús Millán Tejedor.

Disponible en:
http://www.ramonmillan.com/tutoriales/dispositivosviviendadomotica.php#pasarelaresidencial

- [3] "*Hogares Inteligentes …Donde la tecnología y la Arquitectura se unen*",

Disponible en:
http://www.terresdeponent.com/alumnes/Ciencia/Caracteristicas.htm

- [4] "*SMART HOME BLOG*", Meritxell Esquius

Disponible en:
http://www.loxone.com/blog/eses/2013/11/25/ejemplos-domotica-o-automatizacion-asequible/

- [5] "*Manejador de dispositivo*", Wikipedia

Disponible en:
https://es.wikipedia.org/wiki/Manejador_de_dispositivo

- [6] "*DIFERENCIAS ENTRE RASPBERRY PI Y ARDUINO*", Rodríguez Gutiérrez, Adrián

Disponible en:
http://www.conmasfuturo.es/principales-diferencias-entre-raspberry-pi-y-arduino-2/

- [7] "*Android Studio*", wikipedia

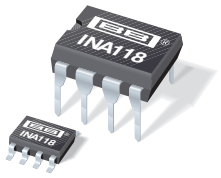
Disponible en:
https://es.wikipedia.org/wiki/Android_Studio

- "*Diseño de un sistema domótico centralizado*", Rodríguez Gutiérrez, Adrián

Disponible en:
https://uvadoc.uva.es/bitstream/10324/12922/1/TFG-P-206.pdf

10. ANEXOS

BURR - BROWN®



INA118

Precision, Low Power INSTRUMENTATION AMPLIFIER

FEATURES

- LOW OFFSET VOLTAGE: 50μV max
- LOW DRIFT: 0.5μV/°C max
- LOW INPUT BIAS CURRENT: 5nA max
- HIGH CMR: 110dB min
- INPUTS PROTECTED TO ±40V
- WIDE SUPPLY RANGE: ±1.35 to ±18V
- LOW QUIESCENT CURRENT: 350μA
- 8-PIN PLASTIC DIP, SO-8

APPLICATIONS

- BRIDGE AMPLIFIER
- THERMOCOUPLE AMPLIFIER
- RTD SENSOR AMPLIFIER
- MEDICAL INSTRUMENTATION
- DATA ACQUISITION

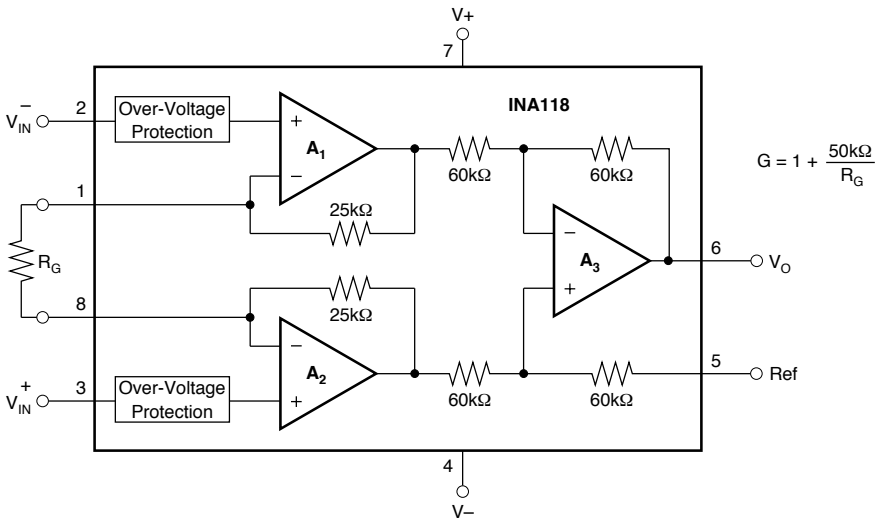
DESCRIPTION

The INA118 is a low power, general purpose instrumentation amplifier offering excellent accuracy. Its versatile 3-op amp design and small size make it ideal for a wide range of applications. Current-feedback input circuitry provides wide bandwidth even at high gain (70kHz at G = 100).

A single external resistor sets any gain from 1 to 10,000. Internal input protection can withstand up to ±40V without damage.

The INA118 is laser trimmed for very low offset voltage (50μV), drift (0.5μV/°C) and high common-mode rejection (110dB at G = 1000). It operates with power supplies as low as ±1.35V, and quiescent current is only 350μA—ideal for battery operated systems.

The INA118 is available in 8-pin plastic DIP, and SO-8 surface-mount packages, specified for the –40°C to +85°C temperature range.



International Airport Industrial Park • Mailing Address: PO Box 11400, Tucson, AZ 85734 • Street Address: 6730 S. Tucson Blvd., Tucson, AZ 85706 • Tel: (520) 746-1111 • Twx: 910-952-1111
Internet: http://www.burr-brown.com/ • FAXLine: (800) 548-6133 (US/Canada Only) • Cable: BBRCORP • Telex: 066-6491 • FAX: (520) 889-1510 • Immediate Product Info: (800) 548-6132

PDS-1199D
Printed in U.S.A. April, 1998
SBOS027

SPECIFICATIONS

ELECTRICAL

At T_A = +25°C, V_S = ±15V, R_L = 10kΩ unless otherwise noted.

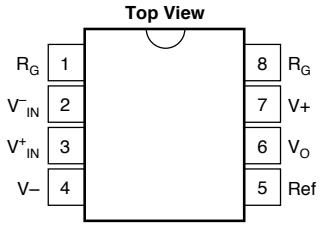
PARAMETER	CONDITIONS	INA118PB, UB			INA118P, U			UNITS
		MIN	TYP	MAX	MIN	TYP	MAX	
INPUT								
Offset Voltage, RTI								
Initial	T_A = +25°C		±10 ± 50/G	±50 ± 500/G		±25 ±100/G	±125±1000/G	µV
vs Temperature	$T_A = T_{MIN}$ to T_{MAX}		±0.2 ± 2/G	±0.5 ± 20/G		±0.2 ± 5/G	±1 ± 20/G	µV/°C
vs Power Supply	V_S = ±1.35V to ±18V		±1 ±10/G	±5 ± 100/G		✻	±10 ±100/G	µV/V
Long-Term Stability			±0.4 ±5/G			✻		µV/mo
Impedance, Differential			10^{10} ‖ 1			✻		Ω ‖ pF
Common-Mode			10^{10} ‖ 4			✻		Ω ‖ pF
Linear Input Voltage Range		(V+) – 1	(V+) – 0.65		✻	✻		V
		(V–) + 1.1	(V–) + 0.95		✻	✻		V
Safe Input Voltage				±40			✻	V
Common-Mode Rejection	V_{CM} = ±10V, ΔR_S = 1kΩ							
	G = 1	80	90		73	✻		dB
	G = 10	97	110		89	✻		dB
	G = 100	107	120		98	✻		dB
	G = 1000	110	125		100	✻		dB
BIAS CURRENT			±1	±5		✻	±10	nA
vs Temperature			±40			✻		pA/°C
OFFSET CURRENT			±1	±5		✻	±10	nA
vs Temperature			±40			✻		pA/°C
NOISE VOLTAGE, RTI	G = 1000, R_S = 0Ω							
f = 10Hz			11			✻		nV/√Hz
f = 100Hz			10			✻		nV/√Hz
f = 1kHz			10			✻		nV/√Hz
f_B = 0.1Hz to 10Hz			0.28			✻		µVp-p
Noise Current								
f=10Hz			2.0			✻		pA/√Hz
f=1kHz			0.3			✻		pA/√Hz
f_B = 0.1Hz to 10Hz			80			✻		pAp-p
GAIN								
Gain Equation			1 + (50kΩ/R_G)			✻		V/V
Range of Gain		1		10000	✻		✻	V/V
Gain Error	G = 1		±0.01	±0.024		✻	±0.1	%
	G = 10		±0.02	±0.4		✻	±0.5	%
	G = 100		±0.05	±0.5		✻	±0.7	%
	G = 1000		±0.5	±1		✻	±2	%
Gain vs Temperature	G = 1		±1	±10		✻	±10	ppm/°C
50kΩ Resistance[1]			±25	±100		✻	✻	ppm/°C
Nonlinearity	G = 1		±0.0003	±0.001		✻	±0.002	% of FSR
	G = 10		±0.0005	±0.002		✻	±0.004	% of FSR
	G = 100		±0.0005	±0.002		✻	±0.004	% of FSR
	G = 1000		±0.002	±0.01		✻	±0.02	% of FSR
OUTPUT								
Voltage: Positive	R_L = 10kΩ	(V+) – 1	(V+) – 0.8		✻	✻		V
Negative	R_L = 10kΩ	(V–) + 0.35	(V–) + 0.2		✻	✻		V
Single Supply High	V_S = +2.7V/0V[2], R_L = 10kΩ	1.8	2.0		✻	✻		V
Single Supply Low	V_S = +2.7V/0V[2], R_L = 10kΩ	60	35		✻	✻		mV
Load Capacitance Stability			1000			✻		pF
Short Circuit Current			+5/–12			✻		mA
FREQUENCY RESPONSE								
Bandwidth, –3dB	G = 1		800			✻		kHz
	G = 10		500			✻		kHz
	G = 100		70			✻		kHz
	G = 1000		7			✻		kHz
Slew Rate	V_O = ±10V, G = 10		0.9			✻		V/µs
Settling Time, 0.01%	G = 1		15			✻		µs
	G = 10		15			✻		µs
	G = 100		21			✻		µs
	G = 1000		210			✻		µs
Overload Recovery	50% Overdrive		20			✻		µs
POWER SUPPLY								
Voltage Range		±1.35	±15	±18	✻	✻	✻	V
Current	V_{IN} = 0V		±350	±385		✻	✻	µA
TEMPERATURE RANGE								
Specification		–40		85	✻		✻	°C
Operating		–40		125	✻		✻	°C
θ_{JA}			80			✻		°C/W

✻ Specification same as INA118PB, UB.

NOTE: (1) Temperature coefficient of the "50kΩ" term in the gain equation. (2) Common-mode input voltage range is limited. See text for discussion of low power supply and single power supply operation.

PIN CONFIGURATION

8-Pin DIP and SO-8



ABSOLUTE MAXIMUM RATINGS

Supply Voltage	±18V
Analog Input Voltage Range	±40V
Output Short-Circuit (to ground)	Continuous
Operating Temperature	–40°C to +125°C
Storage Temperature	–40°C to +125°C
Junction Temperature	+150°C
Lead Temperature (soldering, 10s)	+300°C

ELECTROSTATIC DISCHARGE SENSITIVITY

This integrated circuit can be damaged by ESD. Burr-Brown recommends that all integrated circuits be handled with appropriate precautions. Failure to observe proper handling and installation procedures can cause damage.

ESD damage can range from subtle performance degradation to complete device failure. Precision integrated circuits may be more susceptible to damage because very small parametric changes could cause the device not to meet its published specifications.

ORDERING INFORMATION

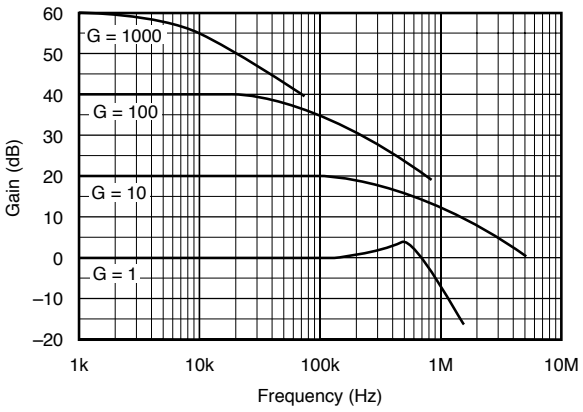
PRODUCT	PACKAGE	PACKAGE DRAWING NUMBER(1)	TEMPERATURE RANGE
INA118P	8-Pin Plastic DIP	006	–40°C to +85°C
INA118PB	8-Pin Plastic DIP	006	–40°C to +85°C
INA118U	SO-8 Surface-Mount	182	–40°C to +85°C
INA118UB	SO-8 Surface-Mount	182	–40°C to +85°C

NOTE: (1) For detailed drawing and dimension table, please see end of data sheet, or Appendix C of Burr-Brown IC Data Book.

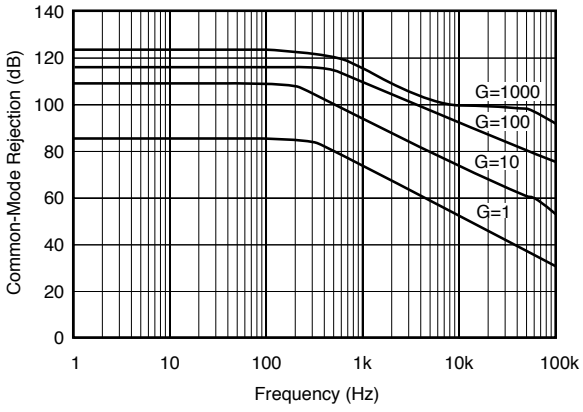
TYPICAL PERFORMANCE CURVES

At T_A = +25°C, V_S = ±15V, unless otherwise noted.

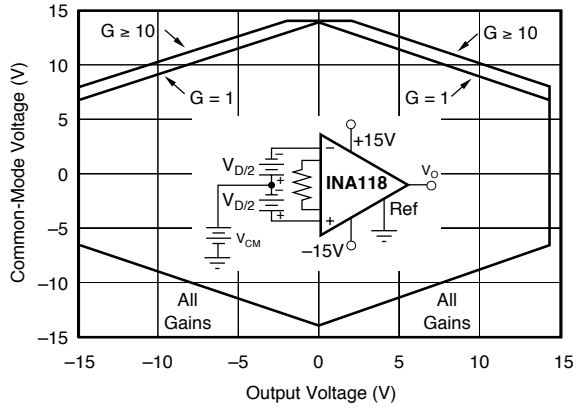
GAIN vs FREQUENCY



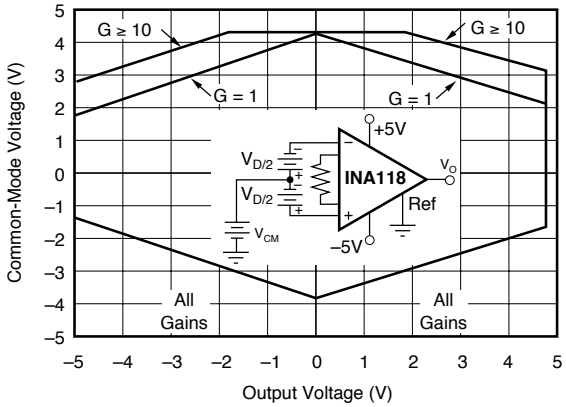
COMMON-MODE REJECTION vs FREQUENCY



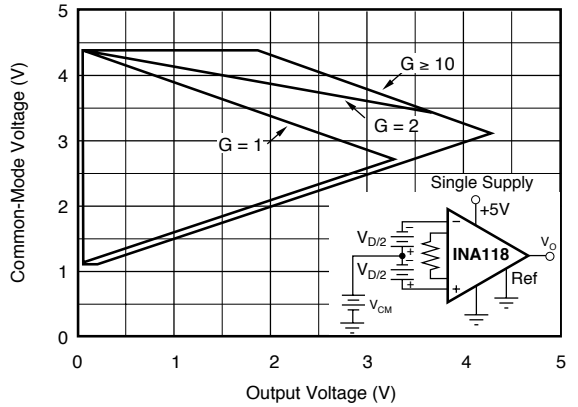
INPUT COMMON-MODE RANGE vs OUTPUT VOLTAGE



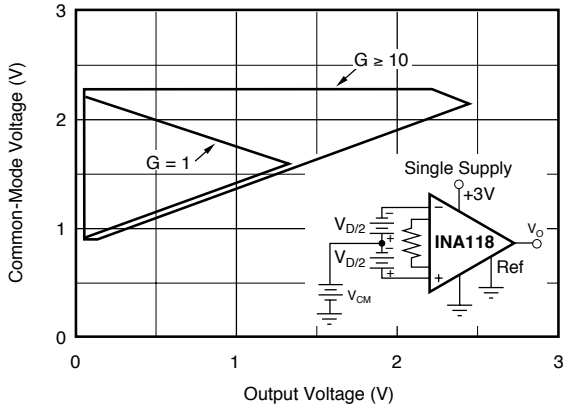
INPUT COMMON-MODE RANGE vs OUTPUT VOLTAGE



INPUT COMMON-MODE RANGE vs OUTPUT VOLTAGE



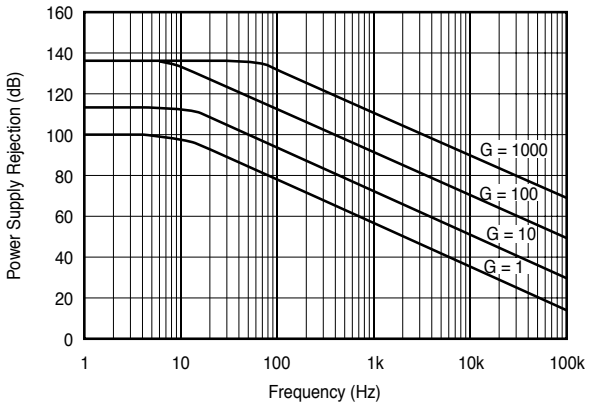
INPUT COMMON-MODE RANGE vs OUTPUT VOLTAGE



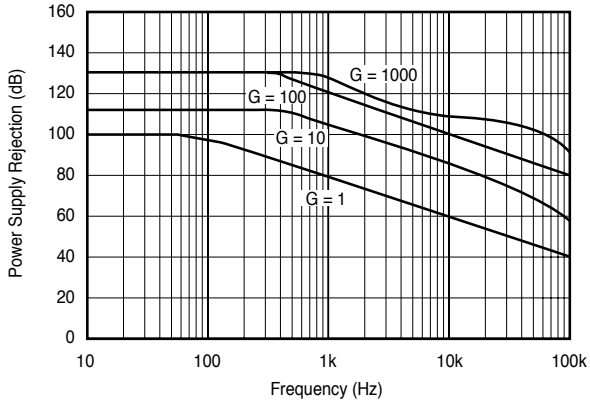
TYPICAL PERFORMANCE CURVES (CONT)

At T_A = +25°C, V_S = ±15V, unless otherwise noted.

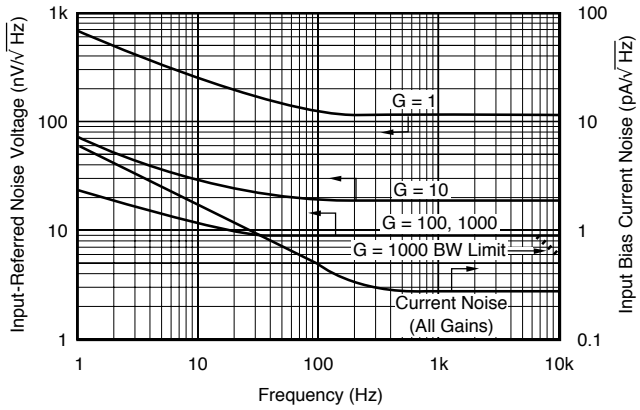
POSITIVE POWER SUPPLY REJECTION vs FREQUENCY



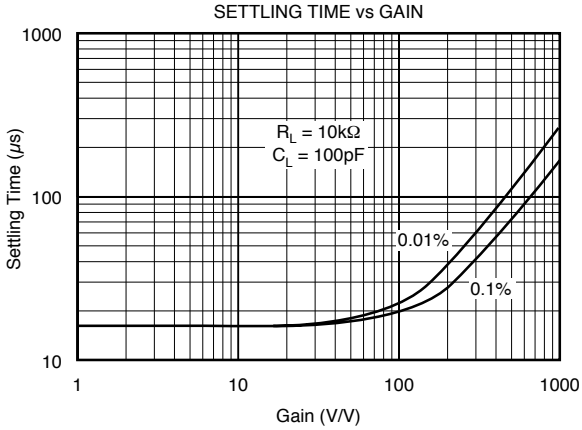
NEGATIVE POWER SUPPLY REJECTION vs FREQUENCY



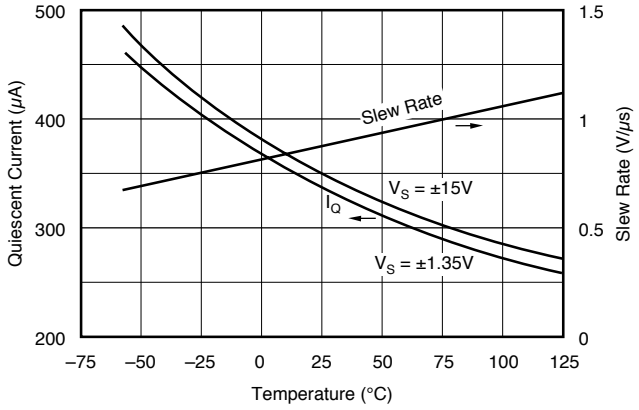
INPUT- REFERRED NOISE VOLTAGE vs FREQUENCY



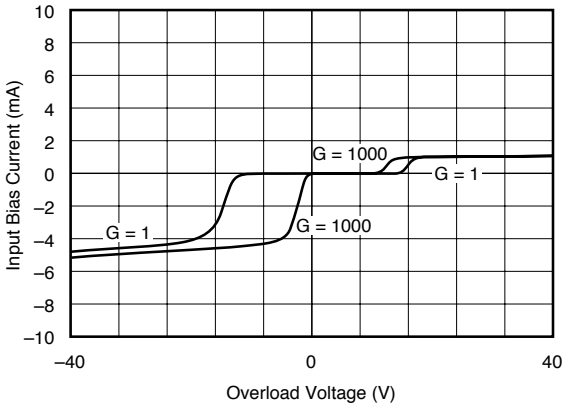
SETTLING TIME vs GAIN



QUIESCENT CURRENT and SLEW RATE vs TEMPERATURE



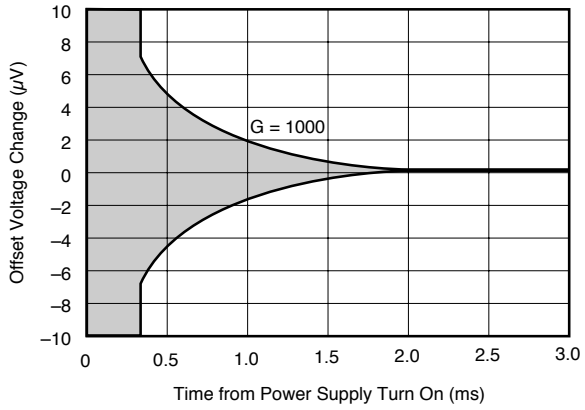
INPUT BIAS CURRENT vs INPUT OVERLOAD VOLTAGE



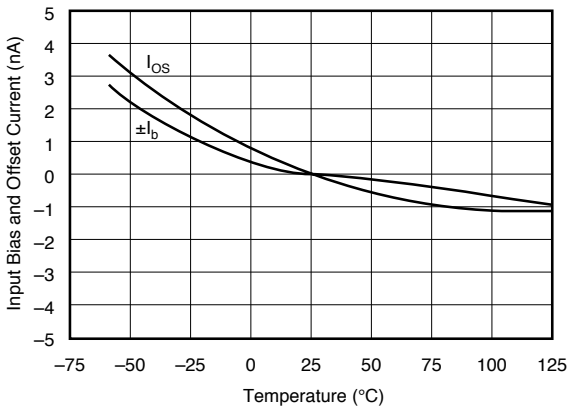
TYPICAL PERFORMANCE CURVES (CONT)

At T_A = +25°C, V_S = ±15V, unless otherwise noted.

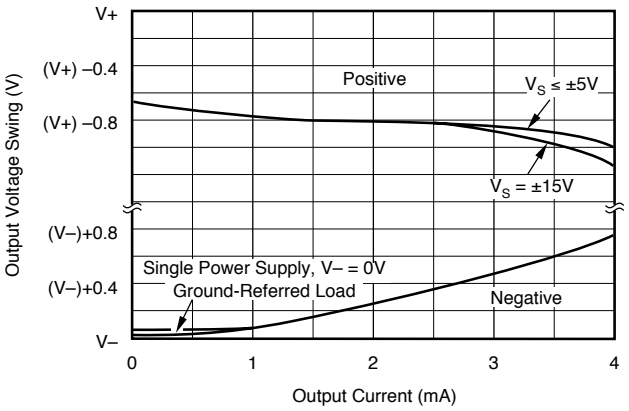
OFFSET VOLTAGE vs WARM-UP TIME



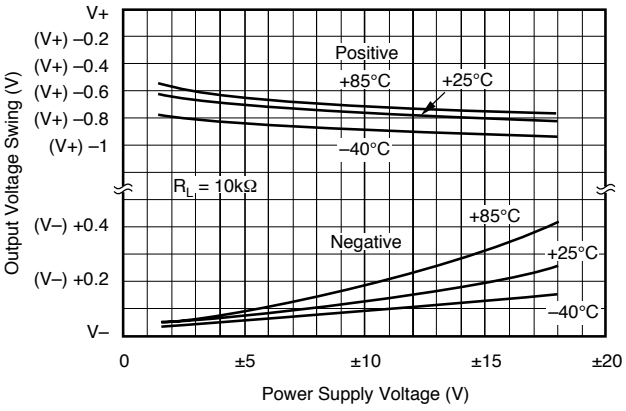
INPUT BIAS AND OFFSET CURRENT vs TEMPERATURE



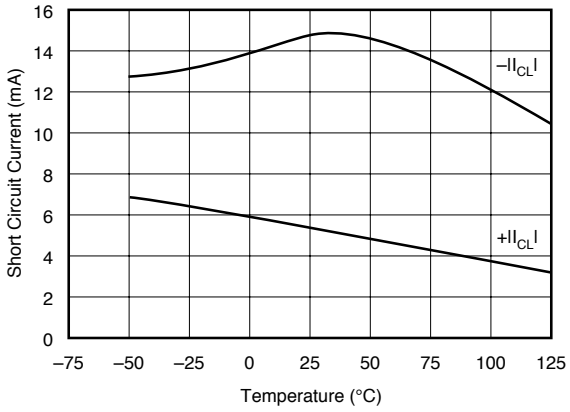
OUTPUT VOLTAGE SWING vs OUTPUT CURRENT



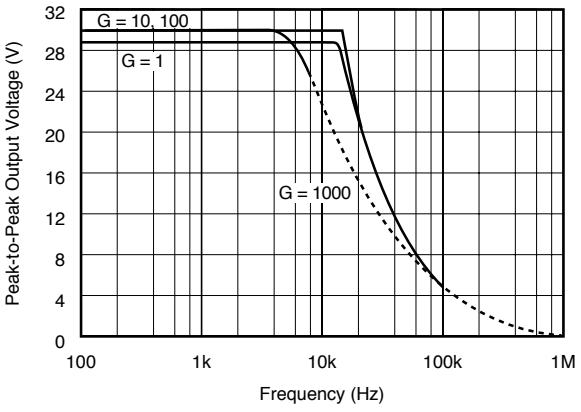
OUTPUT VOLTAGE SWING vs POWER SUPPLY VOLTAGE



OUTPUT CURRENT LIMIT vs TEMPERATURE



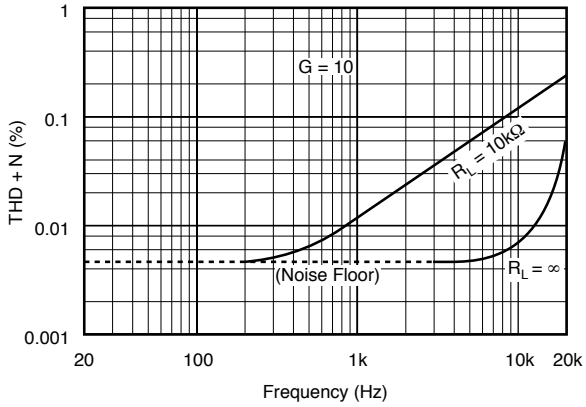
MAXIMUM OUTPUT SWING vs FREQUENCY



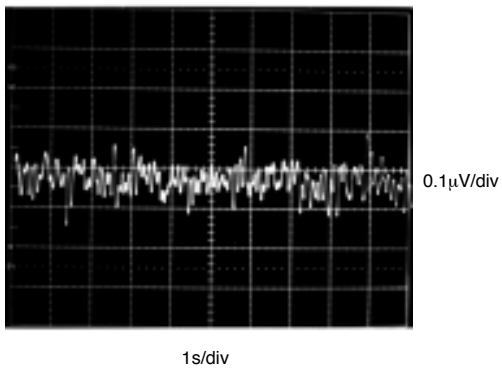
TYPICAL PERFORMANCE CURVES (CONT)

At T_A = +25°C, V_S = ±15V, unless otherwise noted.

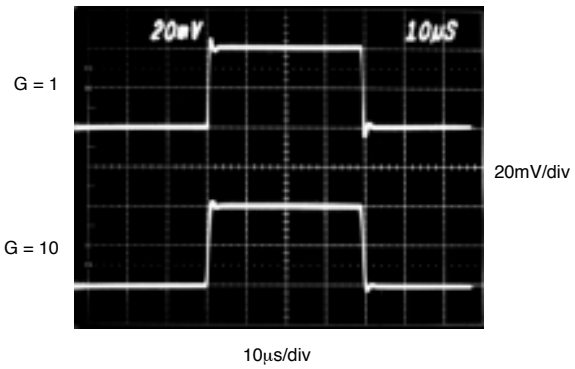
THD + N vs FREQUENCY



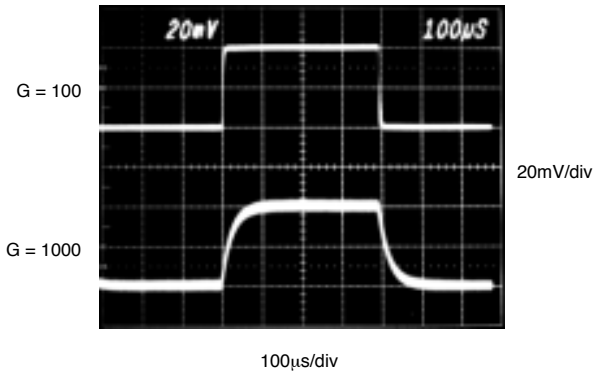
INPUT-REFERRED NOISE, 0.1Hz to 10Hz



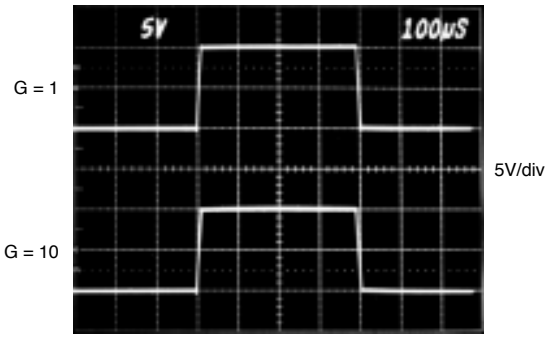
SMALL-SIGNAL RESPONSE



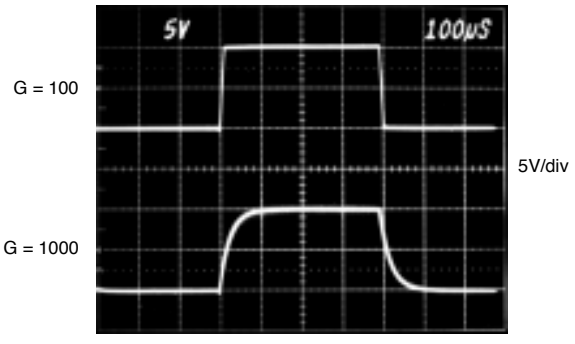
SMALL-SIGNAL RESPONSE



LARGE-SIGNAL RESPONSE



LARGE-SIGNAL RESPONSE



APPLICATION INFORMATION

Figure 1 shows the basic connections required for operation of the INA118. Applications with noisy or high impedance power supplies may require decoupling capacitors close to the device pins as shown.

The output is referred to the output reference (Ref) terminal which is normally grounded. This must be a low-impedance connection to assure good common-mode rejection. A resistance of 12Ω in series with the Ref pin will cause a typical device to degrade to approximately 80dB CMR (G = 1).

SETTING THE GAIN

Gain of the INA118 is set by connecting a single external resistor, R_G, connected between pins 1 and 8:

$$G = 1 + \frac{50k\Omega}{R_G} \quad (1)$$

Commonly used gains and resistor values are shown in Figure 1.

The 50kΩ term in Equation 1 comes from the sum of the two internal feedback resistors of A_1 and A_2. These on-chip metal film resistors are laser trimmed to accurate absolute values. The accuracy and temperature coefficient of these resistors are included in the gain accuracy and drift specifications of the INA118.

The stability and temperature drift of the external gain setting resistor, R_G, also affects gain. R_G's contribution to gain accuracy and drift can be directly inferred from the gain equation (1). Low resistor values required for high gain can make wiring resistance important. Sockets add to the wiring resistance which will contribute additional gain error (possibly an unstable gain error) in gains of approximately 100 or greater.

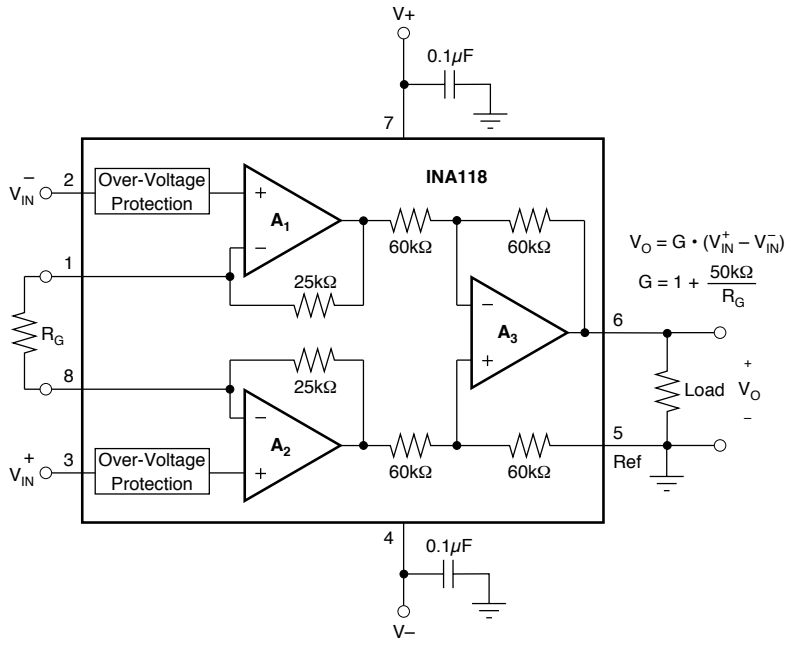
DYNAMIC PERFORMANCE

The typical performance curve "Gain vs Frequency" shows that, despite its low quiescent current, the INA118 achieves wide bandwidth, even at high gain. This is due to the current-feedback topology of the INA118. Settling time also remains excellent at high gain.

The INA118 exhibits approximately 3dB peaking at 500kHz in unity gain. This is a result of its current-feedback topology and is not an indication of instability. Unlike an op amp with poor phase margin, the rise in response is a predictable +6dB/octave due to a response zero. A simple pole at 300kHz or lower will produce a flat passband unity gain response.

DESIRED GAIN	R_G (Ω)	NEAREST 1% R_G (Ω)
1	NC	NC
2	50.00k	49.9k
5	12.50k	12.4k
10	5.556k	5.62k
20	2.632k	2.61k
50	1.02k	1.02k
100	505.1	511
200	251.3	249
500	100.2	100
1000	50.05	49.9
2000	25.01	24.9
5000	10.00	10
10000	5.001	4.99

NC: No Connection.



Also drawn in simplified form:

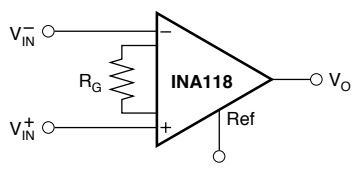


FIGURE 1. Basic Connections.

NOISE PERFORMANCE

The INA118 provides very low noise in most applications. For differential source impedances less than 1kΩ, the INA103 may provide lower noise. For source impedances greater than 50kΩ, the INA111 FET-Input Instrumentation Amplifier may provide lower noise.

Low frequency noise of the INA118 is approximately 0.28μVp-p measured from 0.1 to 10Hz (G≥100). This provides dramatically improved noise when compared to state-of-the-art chopper-stabilized amplifiers.

OFFSET TRIMMING

The INA118 is laser trimmed for low offset voltage and drift. Most applications require no external offset adjustment. Figure 2 shows an optional circuit for trimming the output offset voltage. The voltage applied to Ref terminal is summed at the output. The op amp buffer provides low impedance at the Ref terminal to preserve good common-mode rejection.

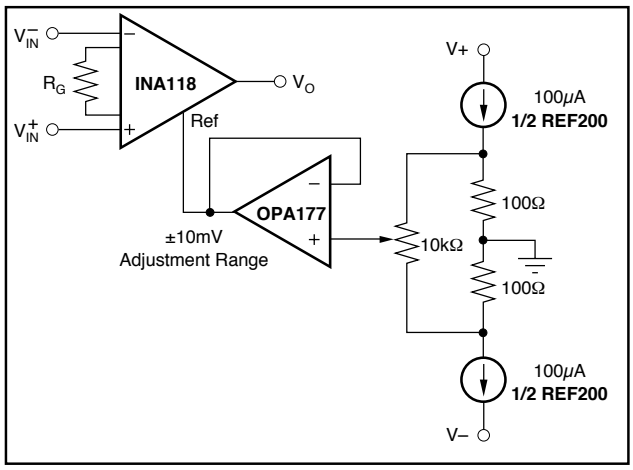


FIGURE 2. Optional Trimming of Output Offset Voltage.

INPUT BIAS CURRENT RETURN PATH

The input impedance of the INA118 is extremely high—approximately 10^{10}Ω. However, a path must be provided for the input bias current of both inputs. This input bias current is approximately ±5nA. High input impedance means that this input bias current changes very little with varying input voltage.

Input circuitry must provide a path for this input bias current for proper operation. Figure 3 shows various provisions for an input bias current path. Without a bias current path, the inputs will float to a potential which exceeds the common-mode range of the INA118 and the input amplifiers will saturate.

If the differential source resistance is low, the bias current return path can be connected to one input (see the thermocouple example in Figure 3). With higher source impedance, using two equal resistors provides a balanced input with possible advantages of lower input offset voltage due to bias current and better high-frequency common-mode rejection.

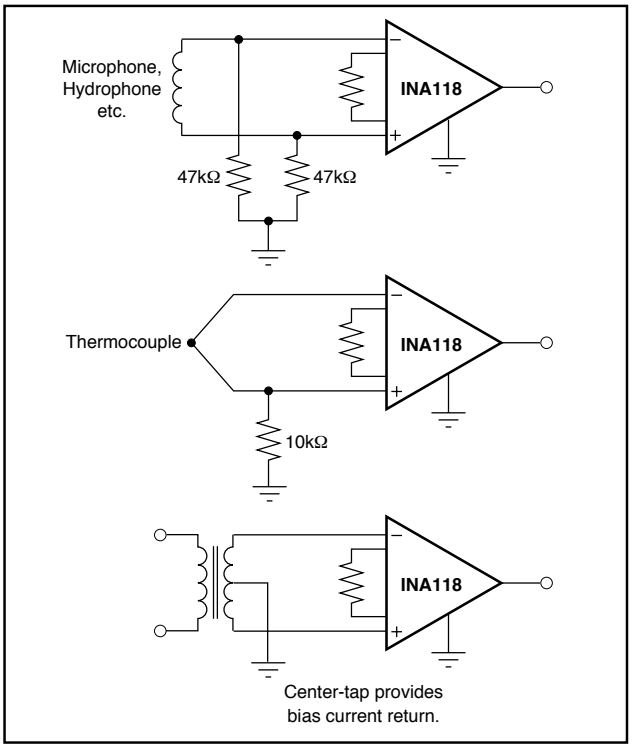


FIGURE 3. Providing an Input Common-Mode Current Path.

INPUT COMMON-MODE RANGE

The linear input voltage range of the input circuitry of the INA118 is from approximately 0.6V below the positive supply voltage to 1V above the negative supply. As a differential input voltage causes the output voltage to increase, however, the linear input range will be limited by the output voltage swing of amplifiers A_1 and A_2. Thus, the linear common-mode input range is related to the output voltage of the complete amplifier. This behavior also depends on supply voltage—see performance curves "Input Common-Mode Range vs Output Voltage".

Input-overload can produce an output voltage that appears normal. For example, if an input overload condition drives both input amplifiers to their positive output swing limit, the difference voltage measured by the output amplifier will be near zero. The output of the INA118 will be near 0V even though both inputs are overloaded.

LOW VOLTAGE OPERATION

The INA118 can be operated on power supplies as low as ±1.35V. Performance of the INA118 remains excellent with power supplies ranging from ±1.35V to ±18V. Most parameters vary only slightly throughout this supply voltage range—see typical performance curves. Operation at very low supply voltage requires careful attention to assure that the input voltages remain within their linear range. Voltage swing requirements of internal nodes limit the input common-mode range with low power supply voltage. Typical performance curves, "Input Common-Mode Range vs Output Voltage" show the range of linear operation for a various supply voltages and gains.

SINGLE SUPPLY OPERATION

The INA118 can be used on single power supplies of +2.7V to +36V. Figure 5 shows a basic single supply circuit. The output Ref terminal is connected to ground. Zero differential input voltage will demand an output voltage of 0V (ground). Actual output voltage swing is limited to approximately 35mV above ground, when the load is referred to ground as shown. The typical performance curve "Output Voltage vs Output Current" shows how the output voltage swing varies with output current.

With single supply operation, V_{IN}^+ and V_{IN}^- must both be 0.98V above ground for linear operation. You cannot, for instance, connect the inverting input to ground and measure a voltage connected to the non-inverting input.

To illustrate the issues affecting low voltage operation, consider the circuit in Figure 5. It shows the INA118, operating from a single 3V supply. A resistor in series with the low side of the bridge assures that the bridge output voltage is within the common-mode range of the amplifier's inputs. Refer to the typical performance curve "Input Common-Mode Range vs Output Voltage" for 3V single supply operation.

INPUT PROTECTION

The inputs of the INA118 are individually protected for voltages up to ±40V. For example, a condition of –40V on one input and +40V on the other input will not cause damage. Internal circuitry on each input provides low series impedance under normal signal conditions. To provide equivalent protection, series input resistors would contribute excessive noise. If the input is overloaded, the protection circuitry limits the input current to a safe value of approximately 1.5 to 5mA. The typical performance curve "Input Bias Current vs Input Overload Voltage" shows this input current limit behavior. The inputs are protected even if the power supplies are disconnected or turned off.

INSIDE THE INA118

Figure 1 shows a simplified representation of the INA118. The more detailed diagram shown here provides additional insight into its operation.

Each input is protected by two FET transistors that provide a low series resistance under normal signal conditions, preserving excellent noise performance. When excessive voltage is applied, these transistors limit input current to approximately 1.5 to 5mA.

The differential input voltage is buffered by Q_1 and Q_2 and impressed across R_G, causing a signal current to flow through R_G, R_1 and R_2. The output difference amp, A_3, removes the common-mode component of the input signal and refers the output signal to the Ref terminal.

Equations in the figure describe the output voltages of A_1 and A_2. The V_{BE} and IR drop across R_1 and R_2 produce output voltages on A_1 and A_2 that are approximately 1V lower than the input voltages.

A_1 Out = $V_{CM} - V_{BE} - (10\mu A \cdot 25k\Omega) - V_O/2$
A_2 Out = $V_{CM} - V_{BE} - (10\mu A \cdot 25k\Omega) + V_O/2$
Output Swing Range A_1, A_2; (V+) – 0.65V to (V–) + 0.06V
Amplifier Linear Input Range: (V+) – 0.65V to (V–) + 0.98V

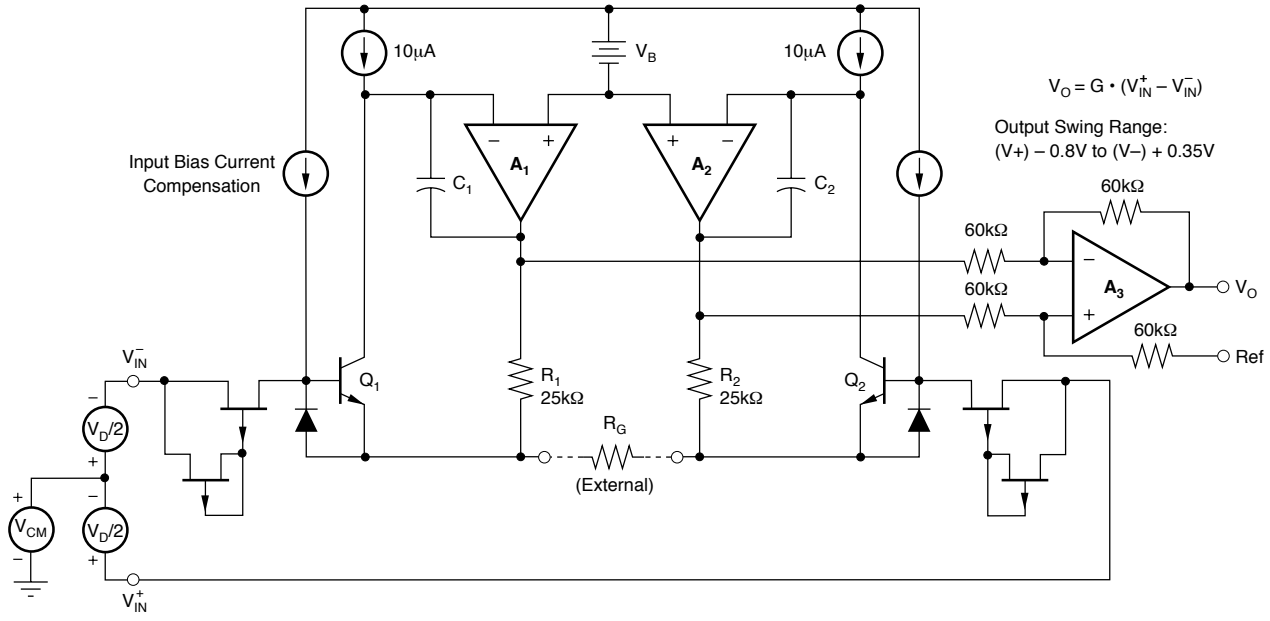


FIGURE 4. INA118 Simplified Circuit Diagram.

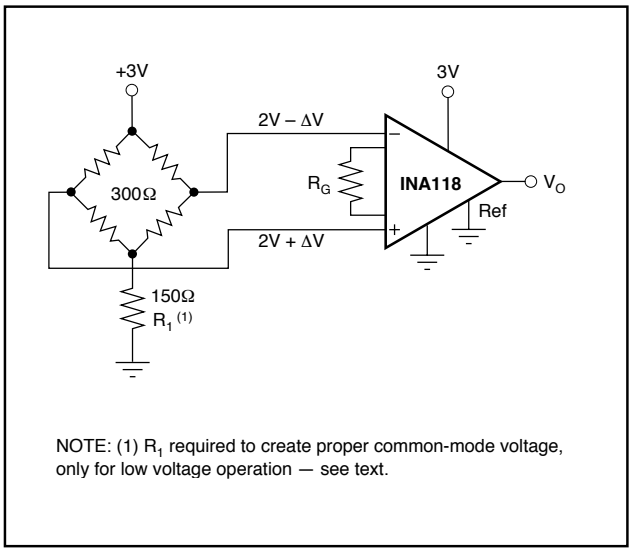


FIGURE 5. Single-Supply Bridge Amplifier.

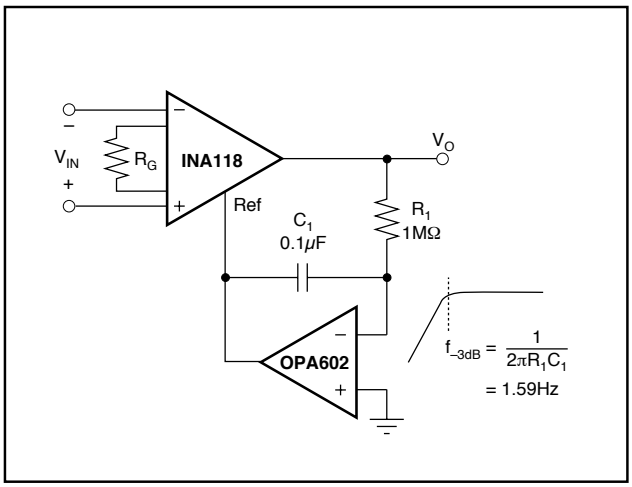
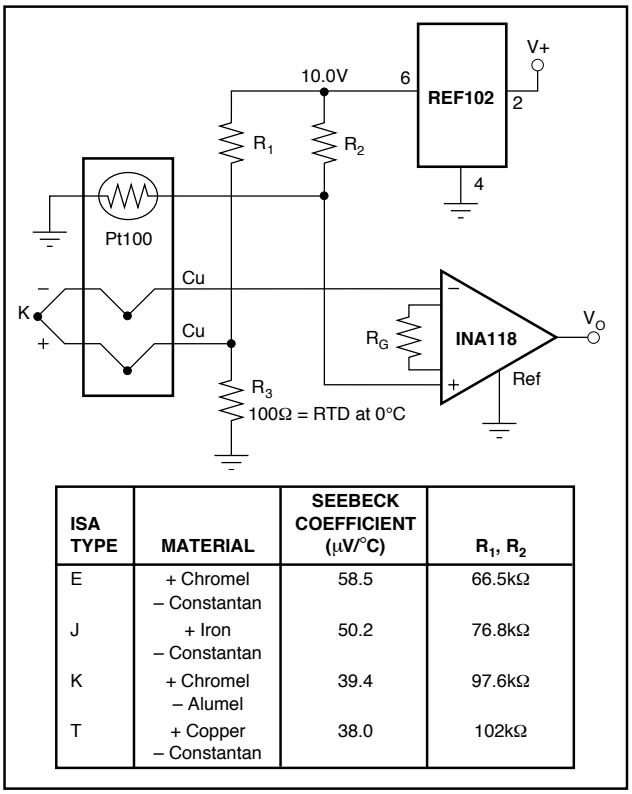
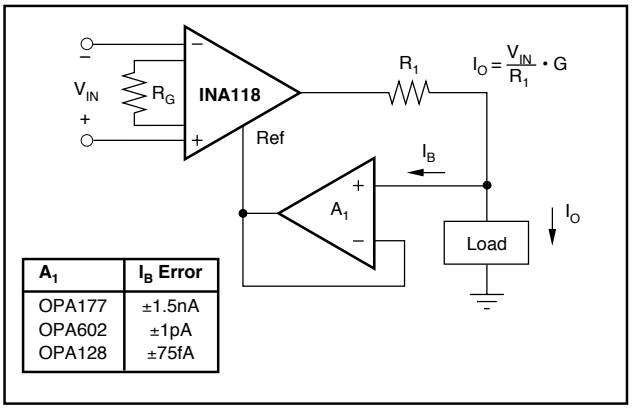


FIGURE 6. AC-Coupled Instrumentation Amplifier.



ISA TYPE	MATERIAL	SEEBECK COEFFICIENT (μV/°C)	R_1, R_2
E	+ Chromel – Constantan	58.5	66.5kΩ
J	+ Iron – Constantan	50.2	76.8kΩ
K	+ Chromel – Alumel	39.4	97.6kΩ
T	+ Copper – Constantan	38.0	102kΩ

FIGURE 7. Thermocouple Amplifier With Cold Junction Compensation.



A_1	I_B Error
OPA177	±1.5nA
OPA602	±1pA
OPA128	±75fA

FIGURE 8. Differential Voltage to Current Converter.

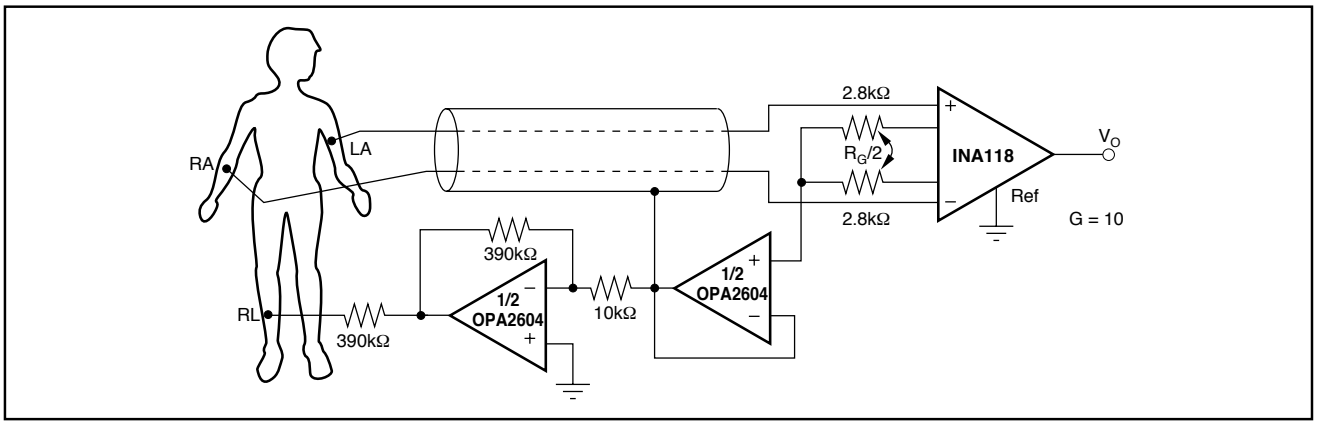


FIGURE 9. ECG Amplifier With Right-Leg Drive.

PACKAGING INFORMATION

Orderable Device	Status (1)	Package Type	Package Drawing	Pins	Package Qty	Eco Plan (2)	Lead/Ball Finish	MSL Peak Temp (3)
INA118P	ACTIVE	PDIP	P	8	50	Green (RoHS & no Sb/Br)	CU NIPDAU	N / A for Pkg Type
INA118PB	ACTIVE	PDIP	P	8	50	Green (RoHS & no Sb/Br)	CU NIPDAU	N / A for Pkg Type
INA118PBG4	ACTIVE	PDIP	P	8	50	Green (RoHS & no Sb/Br)	CU NIPDAU	N / A for Pkg Type
INA118PG4	ACTIVE	PDIP	P	8	50	Green (RoHS & no Sb/Br)	CU NIPDAU	N / A for Pkg Type
INA118U	ACTIVE	SOIC	D	8	75	Green (RoHS & no Sb/Br)	CU NIPDAU	Level-3-260C-168 HR
INA118U/2K5	ACTIVE	SOIC	D	8	2500	Green (RoHS & no Sb/Br)	CU NIPDAU	Level-3-260C-168 HR
INA118U/2K5G4	ACTIVE	SOIC	D	8	2500	Green (RoHS & no Sb/Br)	CU NIPDAU	Level-3-260C-168 HR
INA118UB	ACTIVE	SOIC	D	8	75	Green (RoHS & no Sb/Br)	CU NIPDAU	Level-3-260C-168 HR
INA118UB/2K5	ACTIVE	SOIC	D	8	2500	Green (RoHS & no Sb/Br)	CU NIPDAU	Level-3-260C-168 HR
INA118UB/2K5G4	ACTIVE	SOIC	D	8	2500	Green (RoHS & no Sb/Br)	CU NIPDAU	Level-3-260C-168 HR
INA118UBG4	ACTIVE	SOIC	D	8	75	Green (RoHS & no Sb/Br)	CU NIPDAU	Level-3-260C-168 HR
INA118UG4	ACTIVE	SOIC	D	8	75	Green (RoHS & no Sb/Br)	CU NIPDAU	Level-3-260C-168 HR

(1) The marketing status values are defined as follows:
ACTIVE: Product device recommended for new designs.
LIFEBUY: TI has announced that the device will be discontinued, and a lifetime-buy period is in effect.
NRND: Not recommended for new designs. Device is in production to support existing customers, but TI does not recommend using this part in a new design.
PREVIEW: Device has been announced but is not in production. Samples may or may not be available.
OBSOLETE: TI has discontinued the production of the device.

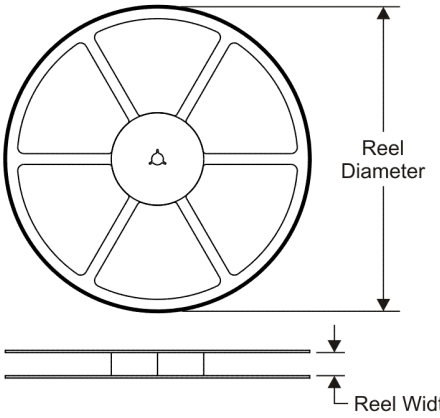
(2) Eco Plan - The planned eco-friendly classification: Pb-Free (RoHS), Pb-Free (RoHS Exempt), or Green (RoHS & no Sb/Br) - please check http://www.ti.com/productcontent for the latest availability information and additional product content details.
TBD: The Pb-Free/Green conversion plan has not been defined.
Pb-Free (RoHS): TI's terms "Lead-Free" or "Pb-Free" mean semiconductor products that are compatible with the current RoHS requirements for all 6 substances, including the requirement that lead not exceed 0.1% by weight in homogeneous materials. Where designed to be soldered at high temperatures, TI Pb-Free products are suitable for use in specified lead-free processes.
Pb-Free (RoHS Exempt): This component has a RoHS exemption for either 1) lead-based flip-chip solder bumps used between the die and package, or 2) lead-based die adhesive used between the die and leadframe. The component is otherwise considered Pb-Free (RoHS compatible) as defined above.
Green (RoHS & no Sb/Br): TI defines "Green" to mean Pb-Free (RoHS compatible), and free of Bromine (Br) and Antimony (Sb) based flame retardants (Br or Sb do not exceed 0.1% by weight in homogeneous material)

(3) MSL, Peak Temp. -- The Moisture Sensitivity Level rating according to the JEDEC industry standard classifications, and peak solder temperature.

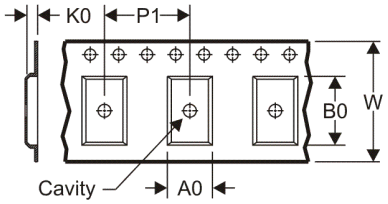
In no event shall TI's liability arising out of such information exceed the total purchase price of the TI part(s) at issue in this document sold by TI to Customer on an annual basis.

TAPE AND REEL INFORMATION

REEL DIMENSIONS

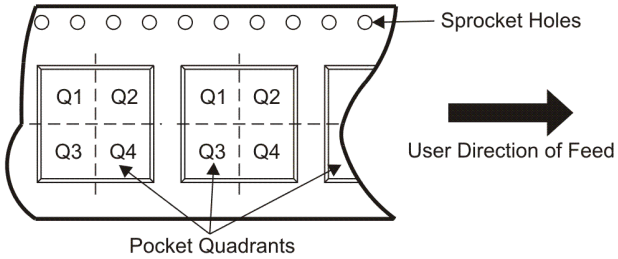


TAPE DIMENSIONS



A0	Dimension designed to accommodate the component width
B0	Dimension designed to accommodate the component length
K0	Dimension designed to accommodate the component thickness
W	Overall width of the carrier tape
P1	Pitch between successive cavity centers

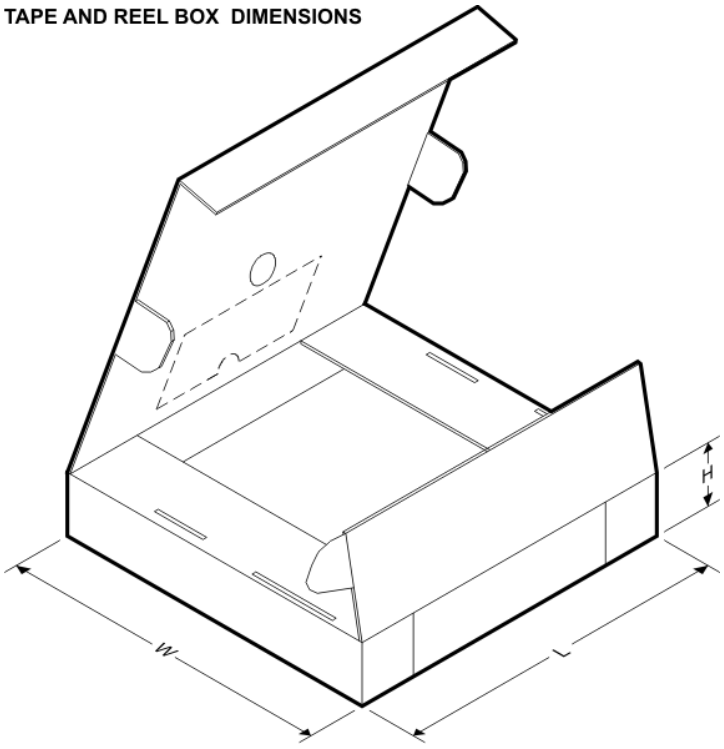
QUADRANT ASSIGNMENTS FOR PIN 1 ORIENTATION IN TAPE



*All dimensions are nominal

Device	Package Type	Package Drawing	Pins	SPQ	Reel Diameter (mm)	Reel Width W1 (mm)	A0 (mm)	B0 (mm)	K0 (mm)	P1 (mm)	W (mm)	Pin1 Quadrant
INA118U/2K5	SOIC	D	8	2500	330.0	12.4	6.4	5.2	2.1	8.0	12.0	Q1
INA118UB/2K5	SOIC	D	8	2500	330.0	12.4	6.4	5.2	2.1	8.0	12.0	Q1

TAPE AND REEL BOX DIMENSIONS



*All dimensions are nominal

Device	Package Type	Package Drawing	Pins	SPQ	Length (mm)	Width (mm)	Height (mm)
INA118U/2K5	SOIC	D	8	2500	346.0	346.0	29.0
INA118UB/2K5	SOIC	D	8	2500	346.0	346.0	29.0

IMPORTANT NOTICE

Texas Instruments Incorporated and its subsidiaries (TI) reserve the right to make corrections, modifications, enhancements, improvements, and other changes to its products and services at any time and to discontinue any product or service without notice. Customers should obtain the latest relevant information before placing orders and should verify that such information is current and complete. All products are sold subject to TI's terms and conditions of sale supplied at the time of order acknowledgment.

TI warrants performance of its hardware products to the specifications applicable at the time of sale in accordance with TI's standard warranty. Testing and other quality control techniques are used to the extent TI deems necessary to support this warranty. Except where mandated by government requirements, testing of all parameters of each product is not necessarily performed.

TI assumes no liability for applications assistance or customer product design. Customers are responsible for their products and applications using TI components. To minimize the risks associated with customer products and applications, customers should provide adequate design and operating safeguards.

TI does not warrant or represent that any license, either express or implied, is granted under any TI patent right, copyright, mask work right, or other TI intellectual property right relating to any combination, machine, or process in which TI products or services are used. Information published by TI regarding third-party products or services does not constitute a license from TI to use such products or services or a warranty or endorsement thereof. Use of such information may require a license from a third party under the patents or other intellectual property of the third party, or a license from TI under the patents or other intellectual property of TI.

Reproduction of TI information in TI data books or data sheets is permissible only if reproduction is without alteration and is accompanied by all associated warranties, conditions, limitations, and notices. Reproduction of this information with alteration is an unfair and deceptive business practice. TI is not responsible or liable for such altered documentation. Information of third parties may be subject to additional restrictions.

Resale of TI products or services with statements different from or beyond the parameters stated by TI for that product or service voids all express and any implied warranties for the associated TI product or service and is an unfair and deceptive business practice. TI is not responsible or liable for any such statements.

TI products are not authorized for use in safety-critical applications (such as life support) where a failure of the TI product would reasonably be expected to cause severe personal injury or death, unless officers of the parties have executed an agreement specifically governing such use. Buyers represent that they have all necessary expertise in the safety and regulatory ramifications of their applications, and acknowledge and agree that they are solely responsible for all legal, regulatory and safety-related requirements concerning their products and any use of TI products in such safety-critical applications, notwithstanding any applications-related information or support that may be provided by TI. Further, Buyers must fully indemnify TI and its representatives against any damages arising out of the use of TI products in such safety-critical applications.

TI products are neither designed nor intended for use in military/aerospace applications or environments unless the TI products are specifically designated by TI as military-grade or "enhanced plastic." Only products designated by TI as military-grade meet military specifications. Buyers acknowledge and agree that any such use of TI products which TI has not designated as military-grade is solely at the Buyer's risk, and that they are solely responsible for compliance with all legal and regulatory requirements in connection with such use.

TI products are neither designed nor intended for use in automotive applications or environments unless the specific TI products are designated by TI as compliant with ISO/TS 16949 requirements. Buyers acknowledge and agree that, if they use any non-designated products in automotive applications, TI will not be responsible for any failure to meet such requirements.

Following are URLs where you can obtain information on other Texas Instruments products and application solutions:

Products		**Applications**	
Amplifiers	amplifier.ti.com	Audio	www.ti.com/audio
Data Converters	dataconverter.ti.com	Automotive	www.ti.com/automotive
DLP® Products	www.dlp.com	Broadband	www.ti.com/broadband
DSP	dsp.ti.com	Digital Control	www.ti.com/digitalcontrol
Clocks and Timers	www.ti.com/clocks	Medical	www.ti.com/medical
Interface	interface.ti.com	Military	www.ti.com/military
Logic	logic.ti.com	Optical Networking	www.ti.com/opticalnetwork
Power Mgmt	power.ti.com	Security	www.ti.com/security
Microcontrollers	microcontroller.ti.com	Telephony	www.ti.com/telephony
RFID	www.ti-rfid.com	Video & Imaging	www.ti.com/video
RF/IF and ZigBee® Solutions	www.ti.com/lprf	Wireless	www.ti.com/wireless

Mailing Address: Texas Instruments, Post Office Box 655303, Dallas, Texas 75265

10.2. Programación App en Android Studio

10.2.1. Java

10.2.1.1. DatosGlobales

```
package com.example.mretxebeste.tfgandroid;

import android.app.Activity;
import android.app.Application;


public class DatosGlobales extends Application {



    private String path = "http://192.168.1.136/arduino/";
    private SincronizadorLucesDesdeArduino sincronizadorLucesDesdeArduino;

    /*LUCES*/

    private boolean cocinaLuces;
    private boolean salonLuces;
    private boolean pasilloLuces;
    private boolean bañoLuces;

    String cocinaLucesURL = "http://192.168.1.46/arduino/digital/2/";
    String salonLucesURL = "http://192.168.1.46/arduino/digital/3";
    String pasilloLucesURL = "http://192.168.1.46/arduino/digital/4";
    String bañoLucesURL = "http://192.168.1.46/arduino/digital/5";


    public boolean getCocinaLuces() {return cocinaLuces;}
    public boolean getSalonLuces(){return salonLuces;}
    public boolean getBañoLuces() {return bañoLuces;}
    public boolean getPasilloLuces() {return pasilloLuces;}


    public void setCocinaLuces(boolean cl) {cocinaLuces = cl;}
    public void setSalonLuces(boolean sl){salonLuces = sl;}
    public void setBañoLuces(boolean bl){bañoLuces = bl;}
    public void setPasilloLuces(boolean pl){pasilloLuces = pl;}


    /*TERMOSTATO*/

    private Integer tCons = 22;
    private Integer tActual = 19;
    private Integer tMax = 30;
    private Integer tMin = 15;

    /*Comienzo*/

    private boolean primeraVez = true;
    public boolean getPrimeraVez(){return primeraVez;}
    public void setPrimeraVez(boolean b){primeraVez = b;}


    public Integer getTMax(){return tMax;}
```

```
    public Integer getTMin(){return tMin;}

    public void setTCons(Integer i){tCons=i;}

    public Integer getTCons(){return tCons;}


    public Integer gettActual(){return tActual;}
    public void settActual(int i){tActual=i;}


    /*LOG IN*/

    private String usuario= "DomoticApp";

    private String contraseña= "DomoticApp";


    public void setUsuario(String s){usuario=s;}

    public String getUsuario(){return usuario;}

    public void setContraseña(String e){contraseña=e;}

    public String getContraseña(){return contraseña;}

    public void realizarPeticion(Activity actividad)
    {
        sincronizadorLucesDesdeArduino = new SincronizadorLucesDesdeArduino(actividad);
        sincronizadorLucesDesdeArduino.doInBackground();
    }

}
```

10.2.1.2. Energía

```
ackage com.example.mretxebeste.tfgandroid;

import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;

public class Energia extends AppCompatActivity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_energia);
    }
}
```

10.2.1.3. Log in

```
package com.example.mretxebeste.tfgandroid;
import android.content.Intent;
import android.graphics.Color;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.Toast;



public class Login extends AppCompatActivity {

    private Button login;
    private EditText Usuario;
    private EditText contraseña;
    private String Us;
    private String Con;


    private Button botonprueba;
    private EditText textoprueba;


    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);


        setContentView(R.layout.activity_login);

        login = (Button) findViewById(R.id.button);
        Usuario = (EditText) findViewById(R.id.usuario);
        contraseña = (EditText) findViewById(R.id.contraseña);

    }

    public void activitylogin(View v) {
        DatosGlobales datosGlobales = (DatosGlobales) getApplicationContext();

        String i = datosGlobales.getUsuario();
        String s = datosGlobales.getContraseña();


        Us = Usuario.getText().toString();
        Con = contraseña.getText().toString();


        if (Us.equals(i)) {

            Usuario.setBackgroundColor(Color.parseColor("#9e9e9e"));
            contraseña.setBackgroundColor(Color.parseColor("#9e9e9e"));

            if (s.equals(Con)) {


                Toast.makeText(Login.this, "Bienvenido " + Us, Toast.LENGTH_LONG).show();
                Intent e = new Intent(getApplicationContext(), Principal.class);
```

```
            startActivity(e);

        } else {

            contraseña.setBackgroundColor(Color.parseColor("#e57373"));
            Toast.makeText(Login.this, "Contraseña incorrecta", Toast.LENGTH_LONG).show();
        }

    } else {

        Usuario.setBackgroundColor(Color.parseColor("#e57373"));
        contraseña.setBackgroundColor(Color.parseColor("#e57373"));

        Toast.makeText(Login.this, "Usuario o contraseña incorrecta", Toast.LENGTH_LONG).show();
    }


}

public void ResetOnclick(View v) {

    Intent e = new Intent(getApplicationContext(), Reset.class);
    startActivity(e);

}

}
```

10.2.1.4. Control de iluminación

```
package com.example.mretxebeste.tfgandroid;



import android.app.Activity;
import android.os.AsyncTask;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.view.View;

import android.widget.CompoundButton;
import android.widget.Switch;
import android.widget.TextView;
import android.widget.Toast;

import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStreamReader;
import java.net.HttpURLConnection;
import java.net.MalformedURLException;
import java.net.URL;
import java.util.Timer;
import java.util.TimerTask;


public class Luces extends AppCompatActivity

{

    private Switch switchCocina;
    private Switch switchSalon;
    private Switch switchBaño;
    private Switch switchPasillo;
    private TextView cocinaOnOff;
    private TextView bañoOnOff;
    private TextView salonOnOff;
    private TextView pasilloOnOff;
    private TextView apagar;
    private Timer timer;



    String cocinaLucesURL = "http://192.168.1.135/arduino/digital/2";
    String salonLucesURL = "http://192.168.1.135/arduino/digital/3";
    String bañoLucesURL = "http://192.168.1.135/arduino/digital/4";
    String pasilloLucesURL = "http://192.168.1.135/arduino/digital/5";



    /*Datos*/
    private boolean cocinaL;
    private boolean salonL;
    private boolean pasilloL;
    private boolean bañoL;


    /*Actualizacióon*/
```

```
String sentencia="";

@Override
protected void onCreate(Bundle savedInstanceState) {



    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_luces);


    final DatosGlobales datosGlobales = (DatosGlobales) getApplicationContext();
    SincronizadorLucesDesdeArduino sincronizadorLucesDesdeArduino = new
SincronizadorLucesDesdeArduino(this);
    sincronizadorLucesDesdeArduino.execute();

    datosGlobales.setPrimeraVez(false);

    /*Se definen las conexiones con la interfaz gráfica*/
    switchCocina = (Switch) findViewById(R.id.LucesCocina);
    switchCocina.setOnCheckedChangeListener(new CompoundButton.OnCheckedChangeListener()
    {
       public void onCheckedChanged(CompoundButton buttonView, boolean isChecked) {
       // do something, the isChecked will be
       // true if the switch is in the On position
          datosGlobales.setCocinaLuces(switchCocina.isChecked());
          if(switchCocina.isChecked())cocinaOnOff.setText("ON");
          else cocinaOnOff.setText("OFF");
          actualizarDatosAlPulsar("cocina");

    }
    });

    switchSalon = (Switch) findViewById(R.id.LucesSalon);
    switchSalon.setOnCheckedChangeListener(new CompoundButton.OnCheckedChangeListener()
    {
       public void onCheckedChanged(CompoundButton buttonView, boolean isChecked) {
          datosGlobales.setSalonLuces(switchSalon.isChecked());
          if(switchSalon.isChecked())salonOnOff.setText("ON");
          else salonOnOff.setText("OFF");
          actualizarDatosAlPulsar("salon");
       }
    });

    switchBaño = (Switch) findViewById(R.id.LucesBaño);
    switchBaño.setOnCheckedChangeListener(new CompoundButton.OnCheckedChangeListener()
    {
       public void onCheckedChanged(CompoundButton buttonView, boolean isChecked) {
          // do something, the isChecked will be
          // true if the switch is in the On position
          datosGlobales.setBañoLuces(switchBaño.isChecked());
          if(switchBaño.isChecked())bañoOnOff.setText("ON");
          else bañoOnOff.setText("OFF");
          actualizarDatosAlPulsar("baño");
       }
    });

    switchPasillo = (Switch) findViewById(R.id.LucesPasillo);
    switchPasillo.setOnCheckedChangeListener(new CompoundButton.OnCheckedChangeListener()
    {
       public void onCheckedChanged(CompoundButton buttonView, boolean isChecked) {
          // do something, the isChecked will be
```

```
            // true if the switch is in the On position
            datosGlobales.setPasilloLuces(switchPasillo.isChecked());

            if(switchPasillo.isChecked())pasilloOnOff.setText("ON");
            else pasilloOnOff.setText("OFF");
            actualizarDatosAlPulsar("pasillo");

        }
    });

    cocinaOnOff = (TextView) findViewById(R.id.cocinaonoff);
    bañoOnOff = (TextView) findViewById(R.id.bañoonoff);
    salonOnOff = (TextView) findViewById(R.id.salononoff);
    pasilloOnOff = (TextView) findViewById(R.id.pasilloonoff);
    apagar = (TextView) findViewById(R.id.apagarluces);


    if(switchCocina.isChecked())cocinaOnOff.setText("ON");
    else cocinaOnOff.setText("OFF");
    if(switchSalon.isChecked()) salonOnOff.setText("ON");
    else salonOnOff.setText("OFF");
    if(switchPasillo.isChecked())pasilloOnOff.setText("ON");
    else pasilloOnOff.setText("OFF");
    if(switchBaño.isChecked())bañoOnOff.setText("ON");
    else bañoOnOff.setText("OFF");


    timer = new Timer();
    timer.scheduleAtFixedRate(new actualizarConArduino(),0,10000);


}


public void actualizarDatosAlPulsar(String cual) {

    DatosGlobales datosGlobales = (DatosGlobales) getApplicationContext();

    if(cual.equals("cocina"))
    {

        datosGlobales.setCocinaLuces(switchCocina.isChecked());
        cocinaL = datosGlobales.getCocinaLuces();

        if (switchCocina.isChecked()) {
            switchCocina.setChecked(true);
            cocinaOnOff.setText("ON");
            sentencia = cocinaLucesURL+"/0";
            EjecutarSentencia es = new EjecutarSentencia(this);
            es.execute();
            System.out.println("Actualizando arduino: cocina on");

        } else {

            switchCocina.setChecked(false);
            cocinaOnOff.setText("OFF");
            sentencia = cocinaLucesURL+"/1";
            EjecutarSentencia es = new EjecutarSentencia(this);
            es.execute();
            System.out.println("Actualizando arduino: cocina off");
        }
```

```
}
else if (cual.equals("baño")) {

   datosGlobales.setBañoLuces(switchBaño.isChecked());
   bañoL = datosGlobales.getBañoLuces();

   if (switchBaño.isChecked()) {
      switchBaño.setChecked(true);
      bañoOnOff.setText("ON");
      sentencia = bañoLucesURL + "/0";
      EjecutarSentencia es = new EjecutarSentencia(this);
      es.execute();
      System.out.println("Actualizando arduino: baño on");
   } else {

      switchBaño.setChecked(false);
      bañoOnOff.setText("OFF");
      sentencia = bañoLucesURL + "/1";
      EjecutarSentencia es = new EjecutarSentencia(this);
      es.execute();
      System.out.println("Actualizando arduino: baño off");
   }
}
else if(cual.equals("salon")) {

   datosGlobales.setSalonLuces(switchSalon.isChecked());
   salonL = datosGlobales.getSalonLuces();


   if (switchSalon.isChecked()) {

      switchSalon.setChecked(true);
      salonOnOff.setText("ON");
      sentencia = salonLucesURL + "/0";
      EjecutarSentencia es = new EjecutarSentencia(this);
      es.execute();
      System.out.println("Actualizando arduino: salón on");
   } else {

      switchSalon.setChecked(false);
      salonOnOff.setText("OFF");

      sentencia = salonLucesURL + "/1";
      EjecutarSentencia es = new EjecutarSentencia(this);
      es.execute();
      System.out.println("Actualizando arduino: salón off");
   }
}
else if (cual.equals("pasillo"))
{

   datosGlobales.setPasilloLuces(switchPasillo.isChecked());
   pasilloL = datosGlobales.getPasilloLuces();

   if (switchPasillo.isChecked()) {

      switchPasillo.setChecked(true);
      pasilloOnOff.setText("ON");
      sentencia = pasilloLucesURL+"/0";
      EjecutarSentencia es = new EjecutarSentencia(this);
      es.execute();
      System.out.println("Actualizando arduino: pasillo on");
```

```
        } else {

            switchPasillo.setChecked(false);
            pasilloOnOff.setText("OFF");
            sentencia = pasilloLucesURL + "/1";
            EjecutarSentencia es = new EjecutarSentencia(this);
            es.execute();
            System.out.println("Actualizando arduino: pasillo off");
        }
    }

}

public void apagarOnClick(View v) {

    if (!switchCocina.isChecked() && !switchBaño.isChecked() && !switchPasillo.isChecked() &&
!switchSalon.isChecked()) {

        Toast.makeText(Luces.this, "No Hay Luces que Apagar", Toast.LENGTH_SHORT).show();

    } else {
        switchCocina.setChecked(false);
        switchSalon.setChecked(false);
        switchPasillo.setChecked(false);
        switchBaño.setChecked(false);

        ApagarTodos at = new ApagarTodos(this);
        at.execute();

        Toast.makeText(Luces.this, "Apagadas Todas Las Luces", Toast.LENGTH_SHORT).show();

    }
}


private class actualizarConArduino extends TimerTask {
    public void run()
    {
        SincronizadorLucesDesdeArduino sincronizadorLucesDesdeArduino = new
SincronizadorLucesDesdeArduino(Luces.this);
        sincronizadorLucesDesdeArduino.execute();
        System.out.println("sincronizando");
    }
}

public class EjecutarSentencia extends AsyncTask<Void,Void,Void> {


    Activity activity;
    String page;
    public EjecutarSentencia(Activity activity){

        this.activity = activity;
        page="";


    }


    @Override
    protected Void doInBackground(Void... params) {
```

```
        InputStreamReader is;
        BufferedReader br;
        try {

            URL url = new URL(sentencia);
            HttpURLConnection peticion = (HttpURLConnection) url.openConnection();

            is = new InputStreamReader(peticion.getInputStream());

            br = new BufferedReader(is);

            this.activity.runOnUiThread(new GUIRun(this.page,this.activity));

        } catch (MalformedURLException e) {
            e.printStackTrace();
        } catch (IOException e) {
            e.printStackTrace();
        }
        return null;
    }

    private class GUIRun implements Runnable{

        String page;
        Activity activity;

        public GUIRun(String page,Activity activity){

            this.page = page;
            this.activity = activity;

        }

       @Override
        public void run()
        {
            System.out.println("SENTENTZIA: " + sentencia);

        }
    }

}

public class ApagarTodos extends AsyncTask<Void,Void,Void> {

    Activity activity;
    String page;
    public ApagarTodos(Activity activity){

        this.activity = activity;
        page="";


    }

    @Override
    protected Void doInBackground(Void... params) {


        InputStreamReader is;
        BufferedReader br;
```

```
        try {
            String laUrl="";
            for(int i=1;i<5;i++) {
                switch(i)
                {
                    case 1: laUrl = cocinaLucesURL;
                        break;
                    case 2: laUrl = salonLucesURL;
                        break;
                    case 3: laUrl = bañoLucesURL;
                        break;
                    case 4: laUrl = pasilloLucesURL;
                        break;
                }
                laUrl = laUrl + "/1";

                URL url = new URL(laUrl);
                HttpURLConnection peticion = (HttpURLConnection) url.openConnection();

                is = new InputStreamReader(peticion.getInputStream());

                br = new BufferedReader(is);
            }
            this.activity.runOnUiThread(new GUIRun(this.page,this.activity));

        } catch (MalformedURLException e) {
            e.printStackTrace();
        } catch (IOException e) {
            e.printStackTrace();
        }
        return null;
    }

    private class GUIRun implements Runnable{

        String page;
        Activity activity;

        public GUIRun(String page,Activity activity){

            this.page = page;
            this.activity = activity;

        }

        @Override
        public void run()
        {
            System.out.println("SENTENTZIA: " + sentencia);

        }
    }

  }

}
```

10.2.1.5. Panel Principal

```
package com.example.mretxebeste.tfgandroid;

import android.content.Intent;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.util.Log;
import android.view.View;
import android.widget.Button;
import android.widget.SeekBar;
import android.widget.Switch;
import android.widget.TextView;
import android.widget.Toast;


public class Principal extends AppCompatActivity implements SeekBar.OnSeekBarChangeListener {

    private Button luces;
    private Button termostato;
    private Button energia;
    private TextView cocina;
    private TextView baño;
    private TextView salon;
    private TextView pasillo;
    private TextView Tconsigna;
    private TextView Tempact;
    private boolean switchCocina;
    private boolean switchSalon;
    private boolean switchBaño;
    private boolean switchPasillo;
    private SeekBar barra;


    /*DATUAK*/

    private boolean cocinaL =false;
    private boolean salonL=true;
    private boolean pasilloL=true;
    private boolean bañoL=true;
    private int minimo;
    private int maximo;
    private int Tcons;
    private int value;



    @Override
    protected void onCreate(Bundle savedInstanceState) {

        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_principal);

        luces = (Button) findViewById(R.id.botonluces);
        termostato = (Button) findViewById(R.id.botonTermostato);
        energia = (Button) findViewById(R.id.botonEnergia);
        cocina = (TextView) findViewById(R.id.cocinaText);
        salon = (TextView) findViewById(R.id.salonText);
        baño = (TextView) findViewById(R.id.bañoText);
        pasillo = (TextView) findViewById(R.id.pasilloText);
```

```
    Tconsigna =(TextView) findViewById(R.id.Tcons);
    Tempact = (TextView) findViewById(R.id.tact);
    barra= (SeekBar) findViewById(R.id.barra);

    ((SeekBar) findViewById(R.id.barra)).setOnSeekBarChangeListener(this);

    DatosGlobales datosGlobales = (DatosGlobales) getApplicationContext();

    minimo = datosGlobales.getTMin();
    maximo = datosGlobales.getTMax();
    Tcons=datosGlobales.getTCons();

    barra.setProgress(Tcons-minimo);



    actualizarDatos();

}

@Override
protected void onResume()
{
    super.onResume();
    actualizarDatos();


}

public void actualizarDatos()
{
    DatosGlobales datosGlobales = (DatosGlobales) getApplicationContext();

    minimo = datosGlobales.getTMin();
    maximo = datosGlobales.getTMax();
    Tcons=datosGlobales.getTCons();
    barra.setMax(maximo-minimo);


    cocinaL = datosGlobales.getCocinaLuces();
    salonL = datosGlobales.getSalonLuces();
    pasilloL = datosGlobales.getPasilloLuces();
    bañoL = datosGlobales.getBañoLuces();

    Integer i = datosGlobales.getTCons();
    String s = i.toString();
    Tconsigna.setText(s);

    Integer e = datosGlobales.gettActual();
    String act = e.toString();
    Tempact.setText(act);


    if(!datosGlobales.getPrimeraVez()) {
        if (cocinaL) cocina.setText("ON");
        else cocina.setText("OFF");

        if (salonL) salon.setText("ON");
        else salon.setText("OFF");

        if (pasilloL) pasillo.setText("ON");
```

```
        else pasillo.setText("OFF");

        if (bañoL) baño.setText("ON");
        else baño.setText("OFF");
    }
}

public void cocinaonoffOnclick (View v) {

    DatosGlobales datosGlobales = (DatosGlobales) getApplicationContext();

   if(cocinaL) {
      switchCocina=false;
      datosGlobales.setCocinaLuces(switchCocina);
       actualizarDatos();
   }
   else  {
      switchCocina=true;
      datosGlobales.setCocinaLuces(switchCocina);
      actualizarDatos();}

}

public void bañoonoffOnclick (View v) {

    DatosGlobales datosGlobales = (DatosGlobales) getApplicationContext();

    if(bañoL) {
       switchBaño=false;
       datosGlobales.setBañoLuces(switchBaño);
       actualizarDatos();
    }
    else  {
       switchBaño=true;
       datosGlobales.setBañoLuces(switchBaño);
       actualizarDatos();}

}
public void pasilloonoffOnclick (View v) {

    DatosGlobales datosGlobales = (DatosGlobales) getApplicationContext();

    if(pasilloL) {
       switchPasillo=false;
       datosGlobales.setPasilloLuces(switchPasillo);
       actualizarDatos();
    }
    else  {
       switchPasillo=true;
       datosGlobales.setPasilloLuces(switchPasillo);
       actualizarDatos();}

}
public void SalononoffOnclick (View v) {

    DatosGlobales datosGlobales = (DatosGlobales) getApplicationContext();

    if(salonL) {
       switchSalon=false;
       datosGlobales.setSalonLuces(switchSalon);
       actualizarDatos();
```

```
        }
        else  {
            switchSalon=true;
            datosGlobales.setSalonLuces(switchSalon);
            actualizarDatos();}

    }

    public void TermostatomasOnclick (View v) {

        DatosGlobales datosGlobales = (DatosGlobales) getApplicationContext();
        Integer i = datosGlobales.getTCons();

        if(i==maximo){Toast.makeText(Principal.this, "Temperatura maxima alcanzada",
Toast.LENGTH_SHORT).show();}
        if (i<maximo){

            i++;
            datosGlobales.setTCons(i);
            actualizarDatos();
        }


    }
    public void TermostatomenosOnclick (View v) {

        DatosGlobales datosGlobales = (DatosGlobales) getApplicationContext();
        Integer i = datosGlobales.getTCons();

        if(i==minimo){Toast.makeText(Principal.this, "Temperatura minima alcanzada",
Toast.LENGTH_SHORT).show();}


        if (i>minimo){

            i--;
            datosGlobales.setTCons(i);
            actualizarDatos();
        }


    }

    public void termostatoOnClick(View v)
    {
        Intent i = new Intent(getApplicationContext(),Termostato.class);
        startActivity(i);

    }
    public void energiaOnClick(View v)
    {
        Intent i = new Intent(getApplicationContext(),Energia.class);
        startActivity(i);
    }

    public void activityluces(View v)
    {
        Intent i = new Intent(getApplicationContext(),Luces.class);
        startActivity(i);
    }
```

```
    @Override
    public void onProgressChanged(SeekBar seekBar, int progress, boolean fromUser) {

        DatosGlobales datosGlobales = (DatosGlobales) getApplicationContext();
        value=datosGlobales.getTCons();

        minimo = datosGlobales.getTMin();
        seekBar.setProgress(value-minimo);

        int e= progress;
        Tcons=e+minimo;

        datosGlobales.setTCons(Tcons);
        actualizarDatos();


    }

    @Override
    public void onStartTrackingTouch(SeekBar seekBar) {


        DatosGlobales datosGlobales = (DatosGlobales) getApplicationContext();

        value=datosGlobales.getTCons();
        maximo=datosGlobales.getTMax();
        minimo=datosGlobales.getTMin();
        actualizarDatos();

    }

    @Override
    public void onStopTrackingTouch(SeekBar seekBar) {

        DatosGlobales datosGlobales = (DatosGlobales) getApplicationContext();
        Integer i = datosGlobales.getTCons();

        if(i==minimo){Toast.makeText(Principal.this, "Temperatura minima alcanzada",
Toast.LENGTH_SHORT).show();}
        if(maximo == i) {
            Toast.makeText(Principal.this, "Temperatura maxima alcanzada",
Toast.LENGTH_SHORT).show();
        }
    }

}
```

10.2.1.6. Reset

```
package com.example.mretxebeste.tfgandroid;

import android.content.Intent;
import android.graphics.Color;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.view.View;
import android.widget.EditText;
import android.widget.Toast;

public class Reset extends AppCompatActivity {

    private EditText Usuario;
    private EditText reset1;
    private EditText reset2;
    private String Us;
    private String  Con1;
    private String Con2;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_reset);

        Usuario = (EditText) findViewById(R.id.editText);
        reset1 = (EditText) findViewById(R.id.editText2);
        reset2 = (EditText) findViewById(R.id.editText3);

    }

    public void ResetearOnclick (View v) {

        DatosGlobales datosGlobales = (DatosGlobales) getApplicationContext();

        String i = datosGlobales.getUsuario();


        Us= Usuario.getText().toString();
        Con1=reset1.getText().toString();
        Con2=reset2.getText().toString();

        if(Us.equals(i)) {

            Usuario.setBackgroundColor(Color.parseColor("#9e9e9e"));
            reset1.setBackgroundColor(Color.parseColor("#9e9e9e"));
            reset2.setBackgroundColor(Color.parseColor("#9e9e9e"));


            if (Con1.equals(Con2)) {

                reset1.setBackgroundColor(Color.parseColor("#9e9e9e"));
                reset2.setBackgroundColor(Color.parseColor("#9e9e9e"));

                datosGlobales.setContraseña(Con1);

                Intent e = new Intent(getApplicationContext(),Login.class);
                startActivity(e);
                Toast.makeText(Reset.this, "Contraseña reseteada", Toast.LENGTH_SHORT).show();
            }
            else{
```

```
            reset1.setBackgroundColor(Color.parseColor("#e57373"));
            reset2.setBackgroundColor(Color.parseColor("#e57373"));

            Toast.makeText(Reset.this, "Las contraseñas deben ser iguales",
Toast.LENGTH_SHORT).show();

        }
    }
    else {

        Toast.makeText(Reset.this, "El usuario introducido no existe. Por favor, introduce un usuario
existente", Toast.LENGTH_LONG).show();
        Usuario.setBackgroundColor(Color.parseColor("#e57373"));
        reset1.setBackgroundColor(Color.parseColor("#9e9e9e"));
        reset2.setBackgroundColor(Color.parseColor("#9e9e9e"));

    }



  }
}
```

10.2.1.7. Sincronizador Luces desde Arduino

```
package com.example.mretxebeste.tfgandroid;

import android.app.Activity;
import android.os.AsyncTask;
import android.widget.Switch;

import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStreamReader;
import java.net.HttpURLConnection;
import java.net.MalformedURLException;
import java.net.URL;


public class SincronizadorLucesDesdeArduino extends AsyncTask<Void,Void,Void>{


    Activity activity;
    String page;
    String cocinaLed ="";
    String salonLed ="";
    String bañoLed="";
    String pasilloLed="";


    public SincronizadorLucesDesdeArduino(Activity activity){

        this.activity = activity;
        this.page = "";

    }


    @Override
    protected Void doInBackground(Void... params) {


        InputStreamReader is;
        BufferedReader br;

        try {

            URL url = new URL("http://192.168.1.135/arduino/actualizar/1");


            HttpURLConnection peticion = (HttpURLConnection) url.openConnection();


            is = new InputStreamReader(peticion.getInputStream());

            br = new BufferedReader(is);

            String linea;
            int zenbatgarrena =0;

            linea = br.readLine();
```

```
        while(linea != null){
            zenbatgarrena++;
            this.page = this.page+linea;
            switch(zenbatgarrena)
            {
                case(1) : cocinaLed=linea;break;
                case(2) : salonLed=linea;break;
                case(3) : bañoLed=linea;break;
                case(4) : pasilloLed=linea;break;
            }

            linea = br.readLine();


        }


        this.activity.runOnUiThread(new GUIRun(this.page,this.activity));


    } catch (MalformedURLException e) {
        e.printStackTrace();
    } catch (IOException e) {
        e.printStackTrace();
    }


    return null;
}

private class GUIRun implements Runnable{

    String page;
    Activity activity;

    public GUIRun(String page,Activity activity){

        this.page = page;
        this.activity = activity;

    }

    @Override
    public void run() {


        /*Cocina*/
        if(cocinaLed.equals("1"))
        {
            ((Switch) this.activity.findViewById(R.id.LucesCocina)).setChecked(false);
            System.out.println("Cocina: itzali");
        }
        else if (cocinaLed.equals("0"))
        {
            ((Switch) this.activity.findViewById(R.id.LucesCocina)).setChecked(true);
            System.out.println("Cocina: encender");
        }
        else System.out.println("Cocina: error");

        /*Salon*/
        if(salonLed.equals("1"))
        {
```

```
            ((Switch) this.activity.findViewById(R.id.LucesSalon)).setChecked(false);
            System.out.println("Salon: itzali");
        }
        else if (salonLed.equals("0"))
        {
            ((Switch) this.activity.findViewById(R.id.LucesSalon)).setChecked(true);
            System.out.println("Salon: encender");
        }
        else System.out.println("Salon: error");

        /*Baño*/
        if(bañoLed.equals("1"))
        {
            ((Switch) this.activity.findViewById(R.id.LucesBaño)).setChecked(false);
            System.out.println("Baño: itzali");
        }
        else if (bañoLed.equals("0"))
        {
            ((Switch) this.activity.findViewById(R.id.LucesBaño)).setChecked(true);
            System.out.println("Baño: encender");
        }
        else System.out.println("Baño: error");

        /*Pasillo*/
        if(pasilloLed.equals("1"))
        {
            ((Switch) this.activity.findViewById(R.id.LucesPasillo)).setChecked(false);
            System.out.println("Pasillo: itzali");
        }
        else if (pasilloLed.equals("0"))
        {
            ((Switch) this.activity.findViewById(R.id.LucesPasillo)).setChecked(true);
            System.out.println("Pasillo: encender");
        }
        else System.out.println("Pasillo: error");
    }

    public String getLehenengoa()
    {
        return cocinaLed;
    }
}

public static void main ()
{
    SincronizadorLucesDesdeArduino p = new SincronizadorLucesDesdeArduino(new Activity());


}
}
```

10.2.1.8. Sincronizador Temperatura desde Arduino

```
package com.example.mretxebeste.tfgandroid;

import android.app.Activity;
import android.os.AsyncTask;
import android.widget.EditText;
import android.widget.Switch;
import android.widget.TextView;

import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStreamReader;
import java.net.HttpURLConnection;
import java.net.MalformedURLException;
import java.net.URL;
public class SincronizadorTemperaturaDesdeArduino extends AsyncTask<Void,Void,Void>{


    Activity activity;
    String page;
    String temperatura;


    public SincronizadorTemperaturaDesdeArduino(Activity activity){

        this.activity = activity;
        this.page = "";

    }


    @Override
    protected Void doInBackground(Void... params) {


        InputStreamReader is;
        BufferedReader br;

        try {

            URL url = new URL("http://192.168.1.135/arduino/actualizar/1");


            HttpURLConnection peticion = (HttpURLConnection) url.openConnection();


            is = new InputStreamReader(peticion.getInputStream());

            br = new BufferedReader(is);

            String linea;
            int zenbatgarrena =0;

            linea = br.readLine();


            while(linea != null){
                zenbatgarrena++;
```

```
            this.page = this.page+linea;
            switch(zenbatgarrena)
            {

               case(6) : temperatura=linea;break;


            }

            linea = br.readLine();


         }
         this.activity.runOnUiThread(new GUIRun(this.page,this.activity));


      } catch (MalformedURLException e) {
         e.printStackTrace();
      } catch (IOException e) {
         e.printStackTrace();
      }


      return null;
   }


   private class GUIRun implements Runnable{

      String page;
      Activity activity;

      public GUIRun(String page,Activity activity){

         this.page = page;
         this.activity = activity;

      }

      @Override
      public void run() {

         Double i = Double.parseDouble(temperatura);

         i = 83.57 - 0.11635 * i;


         Integer inta = i.intValue();
         String s = inta.toString();
         ((TextView) this.activity.findViewById(R.id.Tempactual)).setText(s +"ºC");
         ((EditText) this.activity.findViewById(R.id.temperaturaSincronizada)).setText(s);
      }
   }

   public static void main ()
   {
      SincronizadorTemperaturaDesdeArduino p = new SincronizadorTemperaturaDesdeArduino(new
Activity());
   }
}
```

10.2.1.9. Control de Temperatura

```
package com.example.mretxebeste.tfgandroid;

import android.app.Activity;
import android.graphics.Color;
import android.net.Uri;
import android.os.AsyncTask;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.text.Editable;
import android.text.TextWatcher;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.TextView;
import android.widget.Toast;

import com.google.android.gms.appindexing.Action;
import com.google.android.gms.appindexing.AppIndex;
import com.google.android.gms.common.api.GoogleApiClient;

import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStreamReader;
import java.net.HttpURLConnection;
import java.net.MalformedURLException;
import java.net.URL;
import java.util.Timer;
import java.util.TimerTask;


public class Termostato extends AppCompatActivity {

    private Button mas;
    private TextView Tempact;
    private Button menos;
    private TextView Temperaturaconsigna;
    private EditText TemperaturaSincronizada;
    private Timer timer2;

    private int minimo;
    private int maximo;

    String sentencia="";

    String TACTUALURL = "http://192.168.1.135/arduino/digital/6/";


    private GoogleApiClient client;


    @Override

    protected void onCreate(Bundle savedInstanceState) {
        SincronizadorTemperaturaDesdeArduino stda = new SincronizadorTemperaturaDesdeArduino(this);
        stda.execute();

        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_termostato);
        DatosGlobales dg = (DatosGlobales) getApplicationContext();
```

```
        Tempact = (TextView) findViewById(R.id.Tempactual);

        TemperaturaSincronizada = (EditText) findViewById(R.id.temperaturaSincronizada);


        ////////////////

        TemperaturaSincronizada.addTextChangedListener(new TextWatcher() {

            DatosGlobales dg = (DatosGlobales) getApplicationContext();

            @Override
            public void afterTextChanged(Editable s) {

            }

            @Override
            public void beforeTextChanged(CharSequence s, int start,
            int count, int after) {
        }

            @Override
            public void onTextChanged(CharSequence s, int start,

            int before, int count) {
                String aux = s.toString();
                Integer i = Integer.parseInt(s.toString());
                dg.settActual(i);

            }
        });


        mas = (Button) findViewById(R.id.buttonmas);
        menos = (Button) findViewById(R.id.buttonmenos);
        Temperaturaconsigna = (TextView) findViewById(R.id.TempConsigna);
        minimo = dg.getTMin();
        maximo = dg.getTMax();

        actualizarTCons();


        client = new GoogleApiClient.Builder(this).addApi(AppIndex.API).build();

        System.out.println("TIMER VA!");
        timer2 = new Timer();
        timer2.scheduleAtFixedRate(new TempactualizarConArduino(),0,30000);



    }

    private class TempactualizarConArduino extends TimerTask {

        public void run() {
            SincronizadorTemperaturaDesdeArduino sincronizarTemperaturaDesdeArduino = new
SincronizadorTemperaturaDesdeArduino(Termostato.this);
            sincronizarTemperaturaDesdeArduino.execute();
            System.out.println("sincronizando Temperatura");

            DatosGlobales dg = (DatosGlobales) getApplicationContext();
```

```
        Integer Tactual = dg.gettActual();
        Integer Tconsigna = dg.getTCons();


        System.out.println("TERMOSTATO");
        System.out.println("Tactual: " + Tactual);
        System.out.println("Tconsigna: " + Tconsigna);
        boolean encendido = false;

        if (Tconsigna - 0.5 >= Tactual) {
            encendido = true;
            sentencia = TACTUALURL + "0";
           EjecutarSentencia es = new EjecutarSentencia(Termostato.this);
            es.execute();

        }
        if (Tconsigna + 0.5 <= Tactual) {
            encendido = false;
            sentencia = TACTUALURL + "1";
            EjecutarSentencia es = new EjecutarSentencia(Termostato.this);
            es.execute();


        }
        if (Tconsigna - 0.5 < Tactual && Tactual < Tconsigna + 0.5) {

            if (encendido) {
                sentencia = TACTUALURL + "0";
                EjecutarSentencia es = new EjecutarSentencia(Termostato.this);
                es.execute();


            } else {
                sentencia = TACTUALURL + "1";
                EjecutarSentencia es = new EjecutarSentencia(Termostato.this);
                es.execute();


            }


        }
    }

    }

  public void plusOnClick(View v) {

    DatosGlobales dg = (DatosGlobales) getApplicationContext();
    Integer i = dg.getTCons();


    if (i == maximo) {

        desactivarMas();
        Toast.makeText(Termostato.this, "Temperatura máxima alcanzada",
Toast.LENGTH_SHORT).show();
    }
    if (i < maximo) {
        i++;
        dg.setTCons(i);
```

```
            actualizarTCons();

            if (i == maximo) {

                desactivarMas();
            }
            if (i > minimo) activarMenos();

        }

    }

    public void minusOnClick(View v) {

        DatosGlobales dg = (DatosGlobales) getApplicationContext();
        Integer i = dg.getTCons();

        if (i <= minimo) {

            desactivarMenos();
            Toast.makeText(Termostato.this, "Temperatura minima alcanzada",
Toast.LENGTH_SHORT).show();
        }

        if (i > minimo) {
            i--;
            dg.setTCons(i);
            actualizarTCons();
            if (i == minimo) {
                desactivarMenos();
            }

            if (i < maximo) {
                activarMas();
            }
        }


    }

    public void actualizarTCons() {

        DatosGlobales dg = (DatosGlobales) getApplicationContext();

        minimo = dg.getTMin();
        maximo = dg.getTMax();
        Integer i = dg.getTCons();
        String s = i.toString();
        Temperaturaconsigna.setText(s + "ºC");

        Integer e = dg.gettActual();
        String act = e.toString();
        Tempact.setText(act + "ºC");


        if (i == maximo) desactivarMas();
        if (i == minimo) desactivarMenos();


    }

    public void desactivarMas() {
```

```
    mas.setBackgroundColor(Color.parseColor("#bdbdbd"));
}

public void desactivarMenos() {

    menos.setBackgroundColor(Color.parseColor("#bdbdbd"));
}

public void activarMas() {
    //  mas.setEnabled(true);
    mas.setBackgroundColor(Color.parseColor("#e57373"));
}

public void activarMenos() {

    menos.setBackgroundColor(Color.parseColor("#80d8ff"));
}


@Override
public void onStart() {
    super.onStart();


    client.connect();
    Action viewAction = Action.newAction(
            Action.TYPE_VIEW, // TODO: choose an action type.
            "Termostato Page", // TODO: Define a title for the content shown.
            // TODO: If you have web page content that matches this app activity's content,
            // make sure this auto-generated web page URL is correct.
            // Otherwise, set the URL to null.
            Uri.parse("http://host/path"),
            // TODO: Make sure this auto-generated app deep link URI is correct.
            Uri.parse("android-app://com.example.mretxebeste.tfgandroid/http/host/path")
    );
    AppIndex.AppIndexApi.start(client, viewAction);
}

@Override
public void onStop() {
    super.onStop();

    Action viewAction = Action.newAction(
            Action.TYPE_VIEW, // TODO: choose an action type.
            "Termostato Page", // TODO: Define a title for the content shown.
            // TODO: If you have web page content that matches this app activity's content,
            // make sure this auto-generated web page URL is correct.
            // Otherwise, set the URL to null.
            Uri.parse("http://host/path"),
            // TODO: Make sure this auto-generated app deep link URI is correct.
            Uri.parse("android-app://com.example.mretxebeste.tfgandroid/http/host/path")
    );
    AppIndex.AppIndexApi.end(client, viewAction);
    client.disconnect();
}

public class EjecutarSentencia extends AsyncTask<Void,Void,Void> {

    Activity activity;
    String page;
    public EjecutarSentencia(Activity activity){
```

```
        this.activity = activity;
        page="";
    }

    @Override
    protected Void doInBackground(Void... params) {


        InputStreamReader is;
        BufferedReader br;
        try {

            URL url = new URL(sentencia);
            HttpURLConnection peticion = (HttpURLConnection) url.openConnection();

            is = new InputStreamReader(peticion.getInputStream());

            br = new BufferedReader(is);

            this.activity.runOnUiThread(new GUIRun(this.page,this.activity));

        } catch (MalformedURLException e) {
            e.printStackTrace();
        } catch (IOException e) {
            e.printStackTrace();
        }
        return null;
    }

    private class GUIRun implements Runnable{

        String page;
        Activity activity;

        public GUIRun(String page,Activity activity){

            this.page = page;
            this.activity = activity;

        }

        @Override
        public void run()
        {

            System.out.println("SENTENTZIA: " + sentencia);

        }
    }

  }
}
```

10.2.2. Archivos XML

10.2.2.1. Activity Energía

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:paddingBottom="@dimen/activity_vertical_margin"
    android:paddingLeft="@dimen/activity_horizontal_margin"
    android:paddingRight="@dimen/activity_horizontal_margin"
    android:paddingTop="@dimen/activity_vertical_margin"
    tools:context="com.example.mretxebeste.tfgandroid.Energia"
    android:background="#d3d3d3">

    <TextView
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:text="CONSUMO MES ACTUAL"
        android:id="@+id/textView"
        android:layout_alignParentTop="true"
        android:layout_alignParentLeft="true"
        android:layout_alignParentStart="true"
        android:textSize="20dp"
        android:textColor="#7b7b7b" />

    <TextView
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:text="CONSUMO MES ANTERIOR"
        android:id="@+id/textView2"
        android:textSize="20dp"
        android:layout_marginTop="80dp"
        android:layout_below="@+id/textView"
        android:layout_alignRight="@+id/textView3"
        android:layout_alignEnd="@+id/textView3"
        android:textColor="#7b7b7b" />

    <TextView
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:text="CONSUMO MAXIMO"
        android:id="@+id/textView3"
        android:layout_below="@+id/textView2"
        android:layout_alignParentLeft="true"
        android:layout_alignParentStart="true"
        android:textSize="20dp"
        android:layout_marginTop="80dp"
        android:textColor="#7b7b7b" />

    <TextView
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:text="CONSUMO MINIMO"
        android:id="@+id/textView4"
        android:layout_below="@+id/textView3"
```

```
        android:layout_alignParentLeft="true"
        android:layout_alignParentStart="true"
        android:textSize="20dp"
        android:layout_marginTop="80dp"
        android:textColor="#7b7b7b" />

    <TextView
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:text="1020 Wh"
        android:id="@+id/textView15"
        android:layout_below="@+id/textView"
        android:layout_alignParentLeft="true"
        android:layout_alignParentStart="true"
        android:textColor="#000000"
        android:textSize="50dp"
        android:textAlignment="center"
        android:layout_marginTop="10dp" />

    <TextView
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:text="960 Wh"
        android:id="@+id/textView16"
        android:textColor="#000000"
        android:textSize="50dp"
        android:textAlignment="center"
        android:layout_below="@+id/textView2"
        android:layout_centerHorizontal="true"
        android:layout_marginTop="10dp" />

    <TextView
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:text="1656 Wh"
        android:id="@+id/textView17"
        android:textColor="#000000"
        android:textSize="50dp"
        android:textAlignment="center"
        android:layout_below="@+id/textView3"
        android:layout_centerHorizontal="true"
        android:layout_marginTop="10dp" />

    <TextView
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:text="712 Wh"
        android:id="@+id/textView18"
        android:textColor="#000000"
        android:textSize="50dp"
        android:textAlignment="center"
        android:layout_below="@+id/textView4"
        android:layout_centerHorizontal="true"
        android:layout_marginTop="10dp" />
</RelativeLayout>
```

10.2.2.2. Activity Log In

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:paddingBottom="@dimen/activity_vertical_margin"
    android:paddingLeft="@dimen/activity_horizontal_margin"
    android:paddingRight="@dimen/activity_horizontal_margin"
    android:paddingTop="@dimen/activity_vertical_margin"
    tools:context="com.example.mretxebeste.tfgandroid.Login"
    android:background="#d3d3d3">

    <TextView
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:text="Usuario o e-mail"
        android:id="@+id/usuario1"
        android:textColor="#000000"
        android:textSize="30dp"
        android:textAlignment="center"
        android:textStyle="bold"
        android:layout_marginTop="60dp"
        android:gravity="center" />

    <Button
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:text="Continuar"
        android:id="@+id/button"
        android:onClick="activitylogin"
        android:textColor="#000000"
        android:background="#9e9e9e"
        android:layout_above="@+id/Reset"
        android:layout_alignParentLeft="true"
        android:layout_alignParentStart="true"
        android:layout_marginBottom="20dp"
        android:textColorHint="#bcaaa4" />

    <TextView
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:text="Contraseña"
        android:id="@+id/textView12"
        android:textColor="#000000"
        android:textSize="30dp"
        android:textAlignment="center"
        android:textStyle="bold"
        android:gravity="center"
        android:layout_below="@+id/usuario"
        android:layout_alignParentLeft="true"
        android:layout_alignParentStart="true"
        android:layout_marginTop="50dp" />
```

```
    <EditText
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:ems="10"
        android:id="@+id/usuario"
        android:layout_below="@+id/usuario1"
        android:layout_alignParentLeft="true"
        android:layout_alignParentStart="true"
        android:layout_marginTop="30dp"
        android:hint="domoticapp@unavarra.es"
        android:singleLine="true"
        android:textColor="#757575"
        android:typeface="normal" />

    <EditText
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:ems="10"
        android:id="@+id/contraseña"
        android:layout_below="@+id/textView12"
        android:layout_alignParentLeft="true"
        android:layout_alignParentStart="true"
        android:layout_marginTop="30dp"
        android:inputType="textPassword"
        android:hint="Contraseña"
        android:singleLine="true"
        android:textIsSelectable="false"
        android:textColor="#757575"
        android:typeface="normal" />

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="¿ Olvidaste su Contraseña ?"
        android:id="@+id/Reset"
        android:textColor="#039be5"
        android:onClick="ResetOnclick"
        android:layout_alignParentBottom="true"
        android:layout_centerHorizontal="true"
        android:layout_marginBottom="30dp"
        android:textStyle="bold" />

</RelativeLayout>
```

10.2.2.3. Activity Luces

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:paddingBottom="@dimen/activity_vertical_margin"
    android:paddingLeft="@dimen/activity_horizontal_margin"
    android:paddingRight="@dimen/activity_horizontal_margin"
    android:paddingTop="@dimen/activity_vertical_margin"
    tools:context="com.example.mretxebeste.tfgandroid.Luces"
    android:background="#d3d3d3">

    <ScrollView

        android:layout_width="fill_parent"
        android:layout_height="fill_parent"
        android:id="@+id/scrollView"
        android:layout_centerHorizontal="true">

        <RelativeLayout

            android:layout_width="fill_parent"
            android:layout_height="fill_parent">
            <Switch
                android:layout_width="fill_parent"
                android:layout_height="wrap_content"
                android:text="Cocina"
                android:id="@+id/LucesCocina"
                android:layout_marginTop="20dp"
                android:checked="false"
                android:layout_alignParentTop="true"
                android:layout_alignParentLeft="true"
                android:layout_alignParentStart="true"
                android:onClick="cocinaOnClick"
                android:singleLine="false"/>

            <Switch
                android:layout_width="fill_parent"
                android:layout_height="wrap_content"
                android:text="Salon"
                android:id="@+id/LucesSalon"
                android:layout_marginTop="40dp"
                android:checked="false"
                android:layout_below="@+id/LucesCocina"
                android:layout_alignParentLeft="true"
                android:layout_alignParentStart="true"
                android:onClick="salonOnClick" />

            <Switch
                android:layout_width="fill_parent"
                android:layout_height="wrap_content"
                android:text="Baño"
                android:id="@+id/LucesBaño"
                android:layout_marginTop="40dp"
                android:layout_below="@+id/LucesSalon"
                android:layout_alignParentLeft="true"
                android:layout_alignParentStart="true"
                android:checked="false"
```

```
        android:onClick="bañoOnClick" />

    <Switch
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:text="Pasillo"
        android:id="@+id/LucesPasillo"
        android:layout_below="@+id/LucesBaño"
        android:layout_marginTop="40dp"
        android:layout_alignRight="@+id/LucesBaño"
        android:layout_alignEnd="@+id/LucesBaño"
        android:checked="false"
        android:onClick="pasilloOnClick" />

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:textAppearance="?android:attr/textAppearanceSmall"
        android:id="@+id/cocinaonoff"
        android:layout_below="@+id/LucesCocina"
        android:layout_alignRight="@+id/LucesSalon"
        android:layout_alignEnd="@+id/LucesSalon"
        android:textColor="#000000" />

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:textAppearance="?android:attr/textAppearanceSmall"
        android:id="@+id/salononoff"
        android:layout_below="@+id/LucesSalon"
        android:layout_alignParentRight="true"
        android:layout_alignParentEnd="true"
        android:textColor="#000000" />

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:textAppearance="?android:attr/textAppearanceSmall"
        android:id="@+id/bañoonoff"
        android:layout_below="@+id/LucesBaño"
        android:layout_alignParentRight="true"
        android:layout_alignParentEnd="true"
        android:textColor="#000000" />

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:textAppearance="?android:attr/textAppearanceSmall"
        android:id="@+id/pasilloonoff"
        android:layout_below="@+id/LucesPasillo"
        android:layout_alignParentRight="true"
        android:layout_alignParentEnd="true"
        android:textColor="#000000" />

</RelativeLayout>


</ScrollView>
```

```
    <LinearLayout
        android:orientation="vertical"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignBottom="@+id/scrollView"
        android:layout_alignParentLeft="true"
        android:layout_alignParentStart="false"
        android:layout_gravity="bottom"
        android:clickable="true"
        android:layout_marginTop="310dp">

        <TextView
            android:layout_width="fill_parent"
            android:layout_height="wrap_content"
            android:text="APAGADO GENERAL LUCES"
            android:id="@+id/apagarluces"
            android:layout_alignWithParentIfMissing="false"
            android:textSize="25dp"
            android:autoText="false"
            android:layout_below="@+id/pasilloonoff"
            android:layout_alignParentLeft="true"
            android:layout_alignParentStart="true"
            android:textAlignment="center"
            android:layout_marginTop="0dp"
            android:textStyle="bold"
            android:textColor="#000000" />

        <ImageView
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:id="@+id/imageView"
            android:background="@mipmap/apagado"
            android:layout_marginRight="60dp"
            android:layout_marginTop="30dp"
            android:layout_marginBottom="0dp"
            android:onClick="apagarOnClick"
            android:layout_below="@+id/pasilloonoff"
            android:textAlignment="center"
            android:layout_marginLeft="60dp"
            android:focusable="true" />

    </LinearLayout>


</RelativeLayout>
```

10.2.2.4. Activity Principal

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:paddingBottom="@dimen/activity_vertical_margin"
    android:paddingLeft="@dimen/activity_horizontal_margin"
    android:paddingRight="@dimen/activity_horizontal_margin"
    android:paddingTop="@dimen/activity_vertical_margin"
    tools:context="com.example.mretxebeste.tfgandroid.Principal"
    android:background="#d3d3d3">

    <ScrollView
        android:layout_width="fill_parent"
        android:layout_height="fill_parent">
        <RelativeLayout
            android:layout_width="fill_parent"
            android:layout_height="fill_parent">
            <Button
                android:layout_width="fill_parent"
                android:layout_height="wrap_content"
                android:text="CONTROL DE ILUMINACION"
                android:id="@+id/botonluces"
                android:layout_alignParentTop="true"
                android:layout_centerHorizontal="true"
                android:onClick="activityluces"
                android:background="#9e9e9e"
                android:textSize="20dp"
                android:textColor="#000000"
                android:typeface="sans"
                android:textStyle="bold"
                android:shadowColor="#000000" />

            <Button
                android:layout_width="fill_parent"
                android:layout_height="wrap_content"
                android:text="CONTROL DE TEMPERATURA"
                android:id="@+id/botonTermostato"
                android:onClick="termostatoOnClick"
                android:background="#9e9e9e"
                android:layout_below="@+id/textView8"
                android:layout_alignParentLeft="true"
                android:layout_alignParentStart="true"
                android:layout_marginTop="30dp"
                android:textSize="20dp"
                android:textColor="#000000"
                android:typeface="sans"
                android:textStyle="bold"
                android:shadowColor="#000000"
                android:singleLine="false" />

            <Button
```

```
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:id="@+id/botonEnergia"
    android:onClick="energiaOnClick"
    android:background="#9e9e9e"
    android:gravity="center"
    android:singleLine="false"

    android:text="CONTROL DE ENERGIA"
    android:layout_below="@+id/Tcons"
    android:layout_alignParentLeft="true"
    android:layout_alignParentStart="true"
    android:layout_marginTop="50dp"
    android:textSize="20dp"
    android:textColor="#000000"
    android:typeface="sans"
    android:textStyle="bold"
    android:shadowColor="#000000" />

<TextView
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:textAppearance="?android:attr/textAppearanceSmall"
    android:text="Cocina"
    android:id="@+id/textView5"
    android:layout_below="@+id/botonluces"
    android:layout_alignParentLeft="true"
    android:layout_alignParentStart="true"
    android:textColor="#000000"
    android:layout_marginTop="20dp" />

<TextView
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:textAppearance="?android:attr/textAppearanceSmall"
    android:text="Salon"
    android:id="@+id/textView6"
    android:layout_below="@+id/textView5"
    android:layout_alignParentLeft="true"
    android:layout_alignParentStart="true"
    android:textColor="#000000"
    android:layout_marginTop="20dp" />

<TextView
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:textAppearance="?android:attr/textAppearanceSmall"
    android:text="Baño"
    android:id="@+id/textView7"
    android:layout_below="@+id/textView6"
    android:layout_alignParentLeft="true"
    android:layout_alignParentStart="true"
    android:textColor="#000000"
    android:layout_marginTop="20dp" />

<TextView
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:textAppearance="?android:attr/textAppearanceSmall"
    android:text="Pasillo"
```

```
    android:id="@+id/textView8"
    android:textColor="#000000"
    android:layout_below="@+id/textView7"
    android:layout_alignRight="@+id/textView5"
    android:layout_alignEnd="@+id/textView5"
    android:layout_marginTop="20dp" />

<TextView
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:textAppearance="?android:attr/textAppearanceSmall"
    android:id="@+id/cocinaText"
    android:layout_below="@+id/botonluces"
    android:layout_alignParentRight="true"
    android:layout_alignParentEnd="true"
    android:textColor="#000000"
    android:layout_marginTop="20dp"
    android:clickable="false"
    android:onClick="cocinaonoffOnclick" />

<TextView
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:textAppearance="?android:attr/textAppearanceSmall"
    android:id="@+id/salonText"
    android:layout_below="@+id/cocinaText"
    android:layout_alignParentRight="true"
    android:textColor="#000000"
    android:layout_marginTop="20dp"
    android:onClick="SalononoffOnclick" />

<TextView
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:textAppearance="?android:attr/textAppearanceSmall"
    android:id="@+id/bañoText"
    android:layout_below="@+id/salonText"
    android:layout_alignParentRight="true"
    android:layout_alignParentEnd="true"
    android:textColor="#000000"
    android:layout_marginTop="20dp"
    android:onClick="bañoonoffOnclick" />

<TextView
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:textAppearance="?android:attr/textAppearanceSmall"
    android:id="@+id/pasilloText"
    android:textColor="#000000"
    android:layout_below="@+id/bañoText"
    android:layout_alignParentRight="true"
    android:layout_alignParentEnd="true"
    android:layout_marginTop="20dp"
    android:onClick="pasilloonoffOnclick" />

<TextView
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:id="@+id/Tcons"
    android:textColor="#000000"
```

```
            android:layout_alignTop="@+id/textView9"
            android:layout_alignRight="@+id/botonEnergia"
            android:layout_alignEnd="@+id/botonEnergia" />

        <TextView
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="Temperatura Consigna"
            android:id="@+id/textView9"
            android:textColor="#000000"
            android:layout_below="@+id/textView10"
            android:layout_alignParentLeft="true"
            android:layout_alignParentStart="true"
            android:layout_marginTop="20dp" />

        <TextView
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="Temperatura Actual"
            android:id="@+id/textView10"
            android:textColor="#000000"
            android:layout_below="@+id/botonTermostato"
            android:layout_alignParentLeft="true"
            android:layout_alignParentStart="true"
            android:layout_marginTop="20dp" />

        <TextView
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="New Text"
            android:id="@+id/tact"
            android:textColor="#000000"
            android:layout_below="@+id/botonTermostato"
            android:layout_alignRight="@+id/botonTermostato"
            android:layout_alignEnd="@+id/botonTermostato"
            android:layout_marginTop="20dp" />

        <SeekBar
            android:layout_width="fill_parent"
            android:layout_height="wrap_content"
            android:id="@+id/barra"
            android:layout_below="@+id/textView9"
            android:layout_alignParentLeft="true"
            android:layout_alignParentStart="true"
            android:indeterminate="false"
            android:layout_marginTop="10dp" />
    </RelativeLayout>
</ScrollView>

<EditText
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:id="@+id/temperaturaSincronizada"
    android:layout_alignParentBottom="true"
    android:layout_centerHorizontal="true"
    android:visibility="invisible"/>

</RelativeLayout>
```

10.2.2.5. Activity Termostato

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
   xmlns:tools="http://schemas.android.com/tools"
   android:layout_width="fill_parent"
   android:layout_height="fill_parent"
   android:paddingBottom="@dimen/activity_vertical_margin"
   android:paddingLeft="@dimen/activity_horizontal_margin"
   android:paddingRight="@dimen/activity_horizontal_margin"
   android:paddingTop="@dimen/activity_vertical_margin"
   tools:context="com.example.mretxebeste.tfgandroid.Termostato"
   android:background="#d3d3d3">
<ScrollView
   android:layout_width="fill_parent"
   android:layout_height="fill_parent">
   <RelativeLayout
      android:layout_width="fill_parent"
      android:layout_height="fill_parent">

      <LinearLayout
         android:orientation="vertical"
         android:layout_width="match_parent"
         android:layout_height="match_parent">

         <LinearLayout
            android:orientation="vertical"
            android:layout_width="match_parent"
            android:layout_height="match_parent">

            <TextView
               android:layout_width="fill_parent"
               android:layout_height="wrap_content"
               android:text="TEMPERATURA ACTUAL"
               android:id="@+id/Tactual"
               android:layout_alignParentTop="true"
               android:layout_centerHorizontal="true"
               android:layout_marginLeft="0dp"
               android:layout_marginTop="0dp"
               android:textColor="#000000"
               android:textStyle="bold"
               android:textSize="25dp"
               android:textAlignment="center" />

            <TextView
               android:layout_width="fill_parent"
               android:layout_height="wrap_content"
               android:text="18"
               android:id="@+id/Tempactual"
               android:layout_below="@+id/Tactual"
               android:layout_alignLeft="@+id/buttonmenos"
               android:layout_alignStart="@+id/buttonmenos"
               android:textSize="100dp"
               android:numeric="integer"
               android:textStyle="bold"
               android:textColor="#000000"
               android:layout_centerHorizontal="true"
               android:layout_marginTop="5dp"
               android:layout_gravity="center_horizontal"
               android:textAlignment="center" />
```

```
<TextView
   android:layout_width="fill_parent"
   android:layout_height="wrap_content"
   android:text="TEMPERATURA CONSIGNA"
   android:id="@+id/Tconsigna"
   android:layout_centerVertical="true"
   android:layout_alignParentLeft="true"
   android:layout_alignParentStart="true"
   android:textColor="#000000"
   android:textSize="25dp"
   android:textStyle="bold"
   android:layout_marginTop="15dp"
   android:textAlignment="center" />

<TextView
   android:layout_width="fill_parent"
   android:layout_height="wrap_content"
   android:text="22"
   android:id="@+id/TempConsigna"
   android:numeric="integer"
   android:textColor="#ff0000"
   android:textStyle="bold"
   android:textSize="100dp"
   android:layout_below="@+id/buttonmenos"
   android:layout_centerHorizontal="true"
   android:layout_gravity="center_horizontal"
   android:layout_marginTop="15dp"
   android:textAlignment="center" />

<LinearLayout
   android:orientation="horizontal"
   android:layout_width="wrap_content"
   android:layout_height="wrap_content"
   android:layout_gravity="center"
   android:layout_marginTop="20dp">

   <Button
      android:layout_width="117dp"
      android:layout_height="wrap_content"
      android:text="-"
      android:id="@+id/buttonmenos"
      android:textColor="#000000"
      android:textSize="50dp"
      android:background="#80d8ff"
      android:onClick="minusOnClick"
      android:layout_below="@+id/Tconsigna"
      android:layout_centerHorizontal="true"
      android:layout_gravity="center_horizontal"
      android:textAlignment="center"
      android:longClickable="true" />

   <Button
      android:layout_width="117dp"
      android:layout_height="wrap_content"
      android:text="+"
      android:id="@+id/buttonmas"
      android:textColor="#000000"
      android:textSize="50dp"
      android:background="#e57373"
```

```
                android:layout_below="@+id/TempConsigna"
                android:layout_toRightOf="@+id/Tempactual"
                android:layout_toEndOf="@+id/Tempactual"
                android:onClick="plusOnClick"
                android:longClickable="true" />
            <EditText
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:id="@+id/temperaturaSincronizada"
                android:layout_alignParentBottom="true"
                android:layout_centerHorizontal="true"
                android:visibility="invisible"/>
        </LinearLayout>
    </LinearLayout>

</LinearLayout>

</RelativeLayout>
</ScrollView>

</RelativeLayout>
```

10.2.2.6. Activity Reset

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:paddingBottom="@dimen/activity_vertical_margin"
    android:paddingLeft="@dimen/activity_horizontal_margin"
    android:paddingRight="@dimen/activity_horizontal_margin"
    android:paddingTop="@dimen/activity_vertical_margin"
    tools:context="com.example.mretxebeste.tfgandroid.Reset"
    android:background="#d3d3d3">

    <TextView
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:text="Usuario o e-mail"
        android:id="@+id/textView11"
        android:layout_alignParentTop="true"
        android:layout_alignParentLeft="true"
        android:layout_alignParentStart="true"
        android:textStyle="bold"
        android:textSize="30dp"
        android:textColor="#000000"
        android:textAlignment="center"
        android:gravity="center_horizontal" />

    <EditText
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:id="@+id/editText"
        android:ems="10"
        android:layout_below="@+id/textView11"
        android:layout_alignParentLeft="true"
        android:layout_alignParentStart="true"
        android:layout_marginTop="20dp"
        android:hint="Introduzca usuario o e-mail"
        android:singleLine="true" />

    <TextView
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:text="Nueva Contraseña"
        android:id="@+id/textView13"
        android:textStyle="bold"
        android:textSize="30dp"
        android:textColor="#000000"
        android:textAlignment="center"
        android:gravity="center_horizontal"
        android:layout_below="@+id/editText"
        android:layout_alignParentLeft="true"
        android:layout_alignParentStart="true"
        android:layout_marginTop="40dp" />

    <EditText
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:ems="10"
        android:id="@+id/editText2"
        android:layout_alignParentLeft="true"
```

```
        android:layout_alignParentStart="true"
        android:layout_below="@+id/textView13"
        android:layout_marginTop="20dp"
        android:password="true"
        android:maxLines="1"
        android:hint="Introduzca nueva contraseña"
        android:singleLine="true" />

    <TextView
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:text="Repita Nueva Contraseña"
        android:id="@+id/textView14"
        android:textStyle="bold"
        android:textSize="30dp"
        android:textColor="#000000"
        android:textAlignment="center"
        android:gravity="center_horizontal"
        android:layout_below="@+id/editText2"
        android:layout_alignParentLeft="true"
        android:layout_alignParentStart="true"
        android:layout_marginTop="40dp" />

    <EditText
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:id="@+id/editText3"
        android:layout_alignParentStart="true"
        android:layout_below="@+id/textView14"
        android:layout_alignParentLeft="true"
        android:layout_marginTop="20dp"
        android:password="true"
        android:hint="Vuelva a introducir la contraseña"
        android:singleLine="true"/>

    <Button
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:text="Resetear Contraseña"
        android:id="@+id/Resetear"
        android:layout_below="@+id/editText3"
        android:layout_alignParentLeft="true"
        android:layout_alignParentStart="true"
        android:layout_marginTop="20dp"
        android:background="#9e9e9e"
        android:textColor="#000000"
        android:onClick="ResetearOnclick" />
</RelativeLayout>
```

10.2.3. Manifest

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.example.mretxebeste.tfgandroid">
    <uses-permission android:name="android.permission.INTERNET" />
    <uses-permission android:name="android.permission.ACCESS_NETWORK_STATE" />

    <application

        android:name=".DatosGlobales"
        android:allowBackup="true"
        android:icon="@drawable/appweb"
        android:supportsRtl="true"
        android:theme="@style/AppTheme">
        <activity
            android:name=".Login"
            android:label="DomoticApp">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />
                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
        <activity
            android:name=".Energia"
            android:label="Consumo">
            <meta-data
                android:name="android.support.PARENT_ACTIVITY"
                android:value="com.example.mretxebeste.tfgandroid.Principal" />
        </activity>
        <activity
            android:name=".Termostato"
            android:label="Temperatura">
            <meta-data
                android:name="android.support.PARENT_ACTIVITY"
                android:value="com.example.mretxebeste.tfgandroid.Principal" />
        </activity>
        <activity
            android:name=".Luces"
            android:label="Iluminación">
            <meta-data
                android:name="android.support.PARENT_ACTIVITY"
                android:value="com.example.mretxebeste.tfgandroid.Principal" />
        </activity>
        <activity
            android:name=".Principal"
            android:label="APP ANDROID">

        </activity>
        <activity android:name=".Reset"
            android:label="Resetear Contraseña">

        </activity>

        <meta-data
            android:name="com.google.android.gms.version"
            android:value="@integer/google_play_services_version" />

    </application>

</manifest>
```

10.3. Programación Arduino: Bridge

```
/*
  Arduino Yún Bridge example

  This example for the Arduino Yún shows how to use the
  Bridge library to access the digital and analog pins
  on the board through REST calls. It demonstrates how
  you can create your own API when using REST style
  calls through the browser.

  Possible commands created in this shetch:

  "IPprivadaarduino/arduino/digital/13"     ->
digitalRead(13)
  "IPprivadaarduino/arduino/digital/13/1"   ->
digitalWrite(13, HIGH)
  "IPprivadaarduino/arduino/analog/2/123"   ->
analogWrite(2, 123)
  "IPprivadaarduino/arduino/analog/2"       ->
analogRead(2)
  "IPprivadaarduino/arduino/mode/13/input"  -> pinMode(13,
INPUT)
  "IPprivadaarduino/arduino/mode/13/output" -> pinMode(13,
OUTPUT)

  This example code is part of the public domain

  http://www.arduino.cc/en/Tutorial/Bridge

*/

#include <Bridge.h>
#include <BridgeServer.h>
#include <BridgeClient.h>

// Listen to the default port 5555, the Yún webserver
// will forward there all the HTTP requests you send
BridgeServer server;

void setup() {
  // Bridge startup

  pinMode(13, OUTPUT);
  pinMode(0, OUTPUT);

  pinMode(2, OUTPUT);
  pinMode(3, OUTPUT);
  pinMode(4, OUTPUT);
```

```
  pinMode(5, OUTPUT);
  pinMode(6, OUTPUT);
  pinMode(A4, INPUT);
  pinMode(A1, INPUT);
  pinMode(9, INPUT);



  digitalWrite(13, LOW);
  digitalWrite(0, LOW);
  Bridge.begin();
  digitalWrite(13, HIGH);
  digitalWrite(0, HIGH);


  // Listen for incoming connection only from localhost
  // (no one from the external network could connect)
  server.listenOnLocalhost();
  server.begin();
}

void loop() {


  // Get clients coming from server
  BridgeClient client = server.accept();

  // There is a new client?
  if (client) {
    // Process request
    process(client);
    // Close connection and free resources.
    client.stop();
  }

  delay(30); // Poll every 50ms
}



void process(BridgeClient client) {
  // read the command



  String command = client.readStringUntil('/');
```

```
  // is "digital" command?
  if (command == "digital") {
    digitalCommand(client);
  }

  // is "analog" command?
  if (command == "analog") {
    analogCommand(client);
  }

  // is "mode" command?
  if (command == "mode") {
    modeCommand(client);
  }
  if (command == "actualizar") {
    actualizarCommand(client);
  }



}
void actualizarCommand(BridgeClient client) {

  client.println(digitalRead(2));
  client.println(digitalRead(3));
  client.println(digitalRead(4));
  client.println(digitalRead(5));
  client.println(digitalRead(13));
  client.println(analogRead(A4));
}


void digitalCommand(BridgeClient client) {
  int pin, value;

  // Read pin number
  pin = client.parseInt();

  // If the next character is a '/' it means we have an URL
  // with a value like: "/digital/13/1"
  if (client.read() == '/') {
    value = client.parseInt();
    digitalWrite(pin, value);


  } else {
    value = digitalRead(pin);
```

```
  }



  // Send feedback to client

  client.print(F("Pin D"));
  client.print(pin);
  client.print(F(" set to "));
  client.println(value);

  // Update datastore key with the current pin value
  String key = "D";
  key += pin;
  Bridge.put(key, String(value));
}

void analogCommand(BridgeClient client) {
  int pin, value;

  // Read pin number
  pin = client.parseInt();

  // If the next character is a '/' it means we have an URL
  // with a value like: "/analog/5/120"
  if (client.read() == '/') {
    // Read value and execute command
    value = client.parseInt();
    analogWrite(pin, value);

    // Send feedback to client
    client.print(F("Pin D"));
    client.print(pin);
    client.print(F(" set to analog "));
    client.println(value);

    // Update datastore key with the current pin value
    String key = "D";
    key += pin;
    Bridge.put(key, String(value));
  } else {

    // Read analog pin

    if (pin == 4) {

      // client.print(F("La temperatura es de "));
```

```
      client.print(value);
      client.println(F(" grados centigrados"));
    }
    else {

      value = analogRead(pin);

      // Send feedback to client
      client.print(F("Pin A"));
      client.print(pin);
      client.print(F(" reads analog "));
      client.println(value);
    }
    // Update datastore key with the current pin value
    String key = "A";
    key += pin;
    Bridge.put(key, String(value));
  }
}

void modeCommand(BridgeClient client) {
  int pin;

  // Read pin number
  pin = client.parseInt();

  // If the next character is not a '/' we have a malformed
URL
  if (client.read() != '/') {
    client.println(F("error"));
    return;
  }

  String mode = client.readStringUntil('\r');

  if (mode == "input") {
    pinMode(pin, INPUT);
    // Send feedback to client
    client.print(F("Pin D"));
    client.print(pin);
    client.print(F(" configured as INPUT!"));
    return;
  }

  if (mode == "output") {
    pinMode(pin, OUTPUT);
    // Send feedback to client
    client.print(F("Pin D"));
```

```
    client.print(pin);
    client.print(F(" configured as OUTPUT!"));
    return;
  }

  client.print(F("error: invalid mode "));
  client.print(mode);

}
```